LINO SQUARZA

IL C64 AL LAVORO

INTERFACCIAMENTO PER IL COMMODORE 64

FAENZA EDITRICE - DIVISIONE C.E.L.I.

IL C64 AL LAVORO

Lino SQUARZA

IL C64 AL LAVORO

INTERFACCIAMENTO PER IL COMMODORE 64

FAENZA EDITRICE - DIVISIONE C.E.L.I.

CARATTERI SPECIALI DEI LISTATI BASIC

S	significa pulizia schermo (SHIFT CLR/HOME)
q	significa cursore in basso
r	significa REVERSE ON
R	significa REVERSE OFF
U	significa cursore a destra

Direttore della Collana
Ing. Franco Govoni

Redazione:
40137 BOLOGNA, Via Varthema 60
Tel. 051/391755

Amministrazione e spedizione:
40018 FAENZA (RA), Via P. De Crescenzi, 44
Tel. 0546/663488
Telex 550387 EDITFA I

Via Pier De Crescenzi, 44 - 48018 Faenza (RA)
Stampato nel mese di gennaio 1986
LITOGRAFICA FAENZA s.r.l. - Tel. 0546/620592

INDICE

PREFAZIONE

Il Personal Computer si sta diffondendo in misura sempre piu' capillare in ogni settore dell'attivita' umana coinvolgendo un mumero sempre maggiore di persone che per interessi o per lavoro sono spesso molto lontane dalle sue problematiche sia costruttive che di programmazione. Normalmente il primo impatto e' facilitato dalla reperibilita' di programmi pronti dedicati a precise applicazioni e preparati in modo che per utilizzarli non sia necessario diventare prima esperti di computer. Un esempio classico riguarda la gestione automatizzata delle procedure contabili (fatturazione, contabilita', magazzino, ecc.) che hanno riscosso grande successo anche perche' studiate in modo che un buon ragioniere potesse utilizzarle immediatamente o quasi, senza necessita' di imparare l'elettronica od i linguaggi di programmazione. La diffusione del computer pero' tende a diventare sempre piu' capillare interessando anche l'ambiente domestico dal momento che il suo prezzo e' ormai al livello di un normale elettrodomestico. Per tutti i piccoli personal computer esistono in commercio un grande numero di programmi che permettono di trasformarli in magnifici "video-game" e questo effettivamente rappresenta la prima fase d'uso del calcolatore in ambiente domestico. Dopo pero' un periodo piu' o meno lungo la fase "ludica" comincia ad esaurirsi e ci si trova di fronte ad un oggetto del quale si intuiscono le grandi potenzialita' ma col quale pero' non si sa bene che cosa fare. Purtroppo per molti l'esperienza col computer si ferma a questo punto ed esso ben presto finisce in cantina insieme ai "giocattoli" dimenticati. Per altri invece comincia la parte piu' affascinante dell'avventura informatica e

cioe' quella che permettera' di trasformarlo in un potente strumento da utilizzare nelle mille occasioni offerte dalla vita di ogni giorno. Come ogni strumento si rende pero' necessario a questo punto conoscere meglio le sue vere capacita', il suo modo di lavorare e soprattutto come sia possibile fargli eseguire di volta in volta le funzioni che si rendono necessarie. La conoscenza di un linguaggio di programmazione come il Basic permette di risolvere quasi tutti i problemi di calcolo e di elaborazione di dati una volta che essi siano stati introdotti nel computer nonche' governare le diverse unita' periferiche ad esso collegate come stampanti, dischi, plotter e altro, ma purtroppo molti problemi restano insolubili a questo primo livello. Infatti quando e' necessario intervenire direttamente su apparecchiature esterne, ad esempio accendere o spegnere una lampada o la stufa di riscaldamento oppure misurare la temperatura di una stanza il computer non e' certamente in grado di cavarsela da solo. I circuiti elettronici che costituiscono un normale personal computer dovranno essere completati con altri per trasformarlo in un "sistema di controllo computerizzato",qualcosa cioe' in grado di intervenire direttamente sulla realta' esterna in base ad un programma di lavoro prestabilito. Data la grande varieta' dei processi da controllare, dai piu' semplici ai piu' complessi, sarebbe stato assurdo pretendere che il costruttore lo avesse dotato dei necessari circuiti, interfacce, in grado di risolvere ogni problema. Quasi tutti i costruttori hanno percio' dotato i personal di connettori esterni, denominati Porte, ai quali l'utente potra' collegare le schede elettroniche appositamente studiate per ogni specifica applicazione. Nel seguito ci si occupera' dunque di come realizzare le espansioni hardware del computer per renderlo adatto al controllo di apparecchiature esterne e del modo di

scrivere i programmi per farle funzionare nel modo voluto. Prima pero′ e′ necessario chiarire alcuni concetti fondamentali ed acquisire pratica con le tecniche di programmazione indispensabili per mettere in grado il computer di gestire apparecchiature esterne con la sicurezza e la precisione richieste dall′applicazione che si intende automatizzare

INTRODUZIONE AL CONTROLLO COMPUTERIZZATO

Prima di iniziare una trattazione dettagliata del problema conviene spendere qualche parola per chiarire che cosa si intenda per "CONTROLLO COMPUTERIZZATO". Per qualcuno questa espressione puo' richiamare alla memoria immagini di futuristici robot, fabbriche automatizzate o sofisticati satelliti artificiali. Per altri puo' sembrare qualcosa adatto solo per specialisti del ramo, molto interessante ma completamente al di fuori della normale comprensione. Effettivamente scienziati e progettisti informatici fanno largo uso di computer nell'industria spaziale cosi' come nell'automazione industriale e queste applicazioni sono veramente piuttosto complicate,pero' la stessa espressione puo' anche essere usata per indicare applicazioni molto piu' semplici e familiari ma non per questo meno interessanti e certamente alla portata di molti.

IN CHE COSA CONSISTE UN CONTROLLO COMPUTERIZZATO

L'obiettivo principale di questo libro e' di permettere al lettore di comprendere i fondamenti dell'automazione elettronica al fine di mettere in grado anche persone che non si occupano direttamente di sistemi di controllo in campo scientifico od industriale di sfruttare le grandi potenzialita' del computer per la soluzione dei problemi che si presentano nella loro vita quotidiana. Durante lo sviluppo di tali "domestiche" applicazioni il lettore riuscira' a perdere ogni diffidenza nei riguardi dell'automazione scoprendo come ed in quale misura essa trovi sempre maggiore applicazione in tutti i settori di attivita'. Un altro ambizioso obiettivo e' dimostrare come un controllo computerizzato non debba

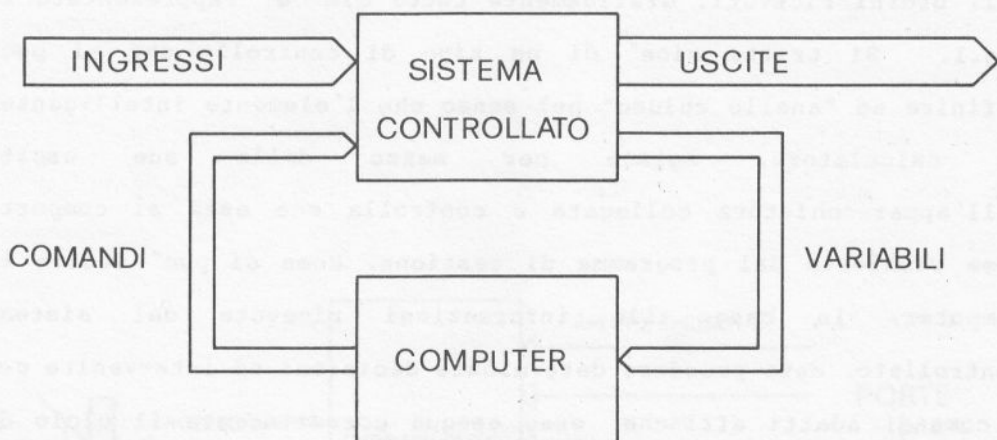



Fig.1 Rappresentazione schematica di un controllo ad "anello chiuso" (Closed Loop)

necessariamente essere complesso ed incomprensibile ad un normale utente di computer. Durante la trattazione verra´ usato un Commodore C64 come unita´ di controllo, ma gli stessi concetti e le stesse tecniche si possono facilmente estendere a molti altri personal computer reperibili sul mercato.

Per chiarire le idee sui controlli computerizzati converra´ ricorrere ad alcuni facili esempi riferiti a situazioni certamente familiari a ciascuno.

In generale si puo´ affermare che in ogni applicazione di questo tipo il computer deve essere in grado di inviare comandi ad una apparecchiatura esterna ed inoltre controllare come essa risponda agli ordini ricevuti. Graficamente tutto cio´ e´ rappresentato in fig.1. Si tratta cioe´ di un tipo di controllo che si puo´ definire ad "anello chiuso" nel senso che l´elemento intelligente, il calcolatore, agisce per mezzo delle sue uscite sull´apparecchiatura collegata e controlla che essa si comporti come richiesto dal programma di gestione. Come si puo´ vedere il computer, in base alle informazioni ricevute dal sistema controllato, deve prendere determinate decisioni ed intervenire con i comandi adatti affinche´ esso esegua correttamente il ciclo di lavoro stabilito in sede di progetto. Questo semplice esempio illustra schematicamente gli elementi elementi fondamentali di un controllo computerizzato e cioe´: invio di comandi all´apparecchiatura esterna e ricezione di informazioni sullo stato della stessa. In conclusione si puo´ affermare che tra il computer e l´esterno deve stabilirsi un flusso di informazioni bidirezionale ed in questo senso ha significato parlare di "dialogo" tra i diversi elementi cosituenti un sistema di controllo computerizzato.

UN ESEMPIO PRATICO DEI CONCETTI BASE

Un esempio di quanto affermato nel capitolo precedente, che

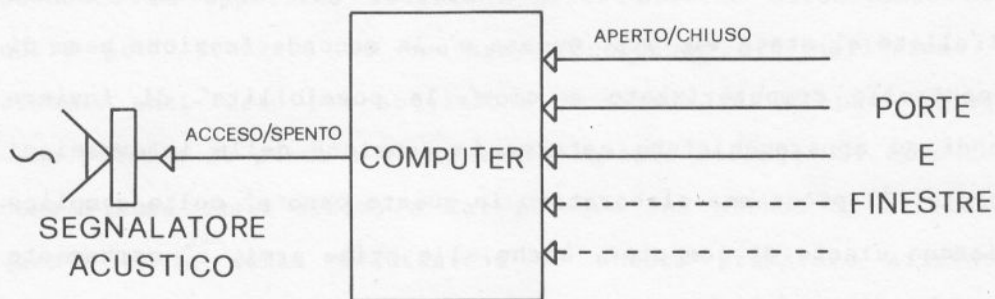



Fig.2 Schematizzazione di un semplice sistema di allarme domestico

certamente e' alla portata di tutti, puo' essere rappresentato da un semplice impianto in grado di impedire l'accesso ad una casa a persone non autorizzate. In pratica si tratta di impedire che un ladro possa introdursi in casa attraverso le porte o le finestre oppure, nel caso che questo avvenga, il sistema deve azionare un dispositivo di allarme acustico o di altro tipo affinche' l'intruso si allontani immediatamente dall'area protetta. Per realizzare cio' e' necessario che ogni possibile via di accesso sia messa in grado di inviare al computer un informazione circa il suo stato in ogni momento. Nel caso specifico ogni porta o finestra deve informare il computer se essa si trova chiusa od aperta. Questo e' il primo requisito di un sistema di controllo computerizzato e cioe' il controllore deve poter conoscere con precisione in ogni momento lo stato del sistema controllato, in questo caso se tutte le possibili vie di accesso alla casa sono aperte o chiuse. Quando si verifica l'apertura di una di esse il computer attivera' una segnalazione acustica esterna, ad esempio una sirena, per avvertire l'area controllata e' stata violata. Questa e' la seconda funzione base di un controllo computerizzato e cioe' la possibilita' di inviare comandi ad apparecchiature esterne in funzione delle informazioni ricevute. Il programma elaborativo in questo caso e' molto semplice e ciascun utente di computer, anche alle prime armi, e' certamente in grado di scriverlo e potrebbe essere sintetizzato nel modo seguente: se almeno una delle aperture e' aperta allora l'allarme deve suonare. Naturalmente si capisce subito che con un tale programma di gestione anche il padrone di casa cadrebbe vittima del suo stesso sistema di sicurezza ogni qual volta rientrasse in casa. Ecco dunque che si avverte immediatamente l'esigenza che l'elaborazione delle informazioni di ingresso sia un po' piu' raffinata, cioe' che il sistema di protezione si comporti in modo

meno banale. E' chiaro che per realizzare quanto sopra non e' assolutamente necessaria la presenza di un computer, basterebbe infatti collegare in parallelo i contatti di protezione sull'alimentazione della sirena di allarme. A questo punto si puo' fare un passo avanti ritardando l'intervento di allarme per il tempo strettamente necessario alla sua disattivazione sapendo esattamente come e dove farlo. In tal modo un estraneo, dovendo perdere tempo nella ricerca del comando di esclusione in tutta la casa, non riuscirebbe mai a raggiungerlo in tempo prima dell'avvio della sirena di allarme. Anche aggiungendo questa possibilita' all'impianto, esso si presta evidentemente a critiche ed a miglioramenti ma l'obiettivo a questo punto e' solo quello di mostrare quali siano gli elementi indispensabili per realizzare un sistema di controllo. Il controllore dunque non deve solo ricevere informazioni sullo stato del sistema controllato e potergli inviare comandi, ma deve anche possedere un programma in base al quale prendere le opportune decisioni nelle diverse circostanze nelle quali puo' trovarsi ad operare. Procedendo ora nell'esame del sistema anti-furto per abitazione domestica, ci si puo' chiedere come possano le porte e le finestre inviare messaggi al computer e come riesca quest'ultimo a far suonare una sirena di allarme. A questo punto della trattazione e' ancora prematuro affrontare questi aspetti del problema, per altro di grande importanza, che verranno pero' ampiamente esaminati e chiariti nel seguito. Quello che preme a questo punto e' comprendere che tutta l'attivita' di un sistema di controllo si basa sulla ripetizione continua delle tre operazioni fondamentali viste e cioe' ricezione delle informazioni dall'esterno, loro elaborazione e invio di comandi alle apparecchiature controllate in base ad un preciso programma di lavoro. Prima di esaminare come possa fisicamente realizzarsi

questo dialogo e' indispensabile chiarire il significato di alcuni concetti che ricorreranno molto frequentemente nel seguito.

NOZIONI FONDAMENTALI PER

IL CONTROLLO COMPUTERIZZATO

GRANDEZZE ANALOGICHE E DIGITALI

La distinzione tra grandezze analogiche e digitali e' di fondamentale importanza per la comprensione e la realizzazione di sistemi di controllo computerizzati dai piu' semplici ai piu' complessi. Si prenda ad esempio una grandezza a tutti familiare: la temperatura. Il suo valore si puo' esprimenre con un numero ,intero o con decimali, che in una certa scala (centigradi, Fahrenheit, Kelvin, ecc.) permette la definisce in modo univoco. Si puo' dunque affermare che una grandezza fisica e' associabile secondo determinate convenzioni (scale) ad un numero che ne rappresenta con l'approssimazione voluta (numero di decimali) il valore vero. Poiche' la temperatura puo' assumere infiniti valori esisteranno altrettanti numeri in grado di rappresentarli. Orbene si puo' affermare che la temperatura e' una grandezza analogica in quanto puo' assumere molti valori, in teoria infiniti. Si pensi ora come ulteriore esempio ad uno strumento per rilevare la temperatura corporea che pero' ci dica solo se il paziente ha la febbre o no accendendo ad esempio una lampada spia in caso affermativo o mantenendola spenta in caso negativo. Si tratta ancora evidentemente di un termometro in grado di misurare la tempertura ad esempio da 20 a 50 gradi centigradi ma la sua indicazione non e' piu' analogica in quanto l'uscita puo' assumere solo due valori diversi (lampada accesa o spenta) a seconda che il valore sia

superiore ai 37 gradi o meno. In tutti i casi in cui l'esito di una misura puo' assumere solo due valori (si o no) e non infiniti (la temperatura corporea) la grandezza si dice digitale. In effetti le grandezze fisiche come temperatura, pressione, umidita', luminosita e molte altre sono intrinsecamente analogiche, cioe' ad infiniti valori, ma possono essere anche misurate in modo digitale ponendo una soglia e controllando solo il suo superamento o meno. Quindi il termine analogico o digitale non si riferisce tanto alla natura di una grandezza ma al modo che si sceglie per misurarla. Vi sono pero' anche esempi di fenomeni puramenti digitali. Ad esempio volendo controllare se una porta e' chiusa od aperta per mezzo di un microinterruttore inserito nello stipite i casi possibili sono solo due e cioe': contatto chiuso significa porta chiusa mentre contatto aperto porta aperta. Si puo' osservare a questo punto che una informazione di tipo digitale permette di decidere tra un numero finito di alternative, nel nostro caso tra due, mentre una digitale tra infinite. Nel seguito i due possibili stati di una grandezza digitale verranno individuati con le due cifre 0 e 1 come e' usuale in campo elettronico ed informatico.

I TRASDUTTORI

Un computer puo' ricevere e trasmettere solo segnali elettrici quindi le grandezze che si vogliono rilevare dovranno per prima cosa essere trasformate in una tensioni o correnti ad esse proporzionali. Analogamente quando dovra' intervenire su di una apparecchiatura controllata il segnale elettrico in uscita dall'elaboratore dovra' essere trasformato nella corrispondente azione,ad esempio l'avviamento di un motore o l'apertura di una valvola od altro. Questi dispositivi in grado di far comunicare il computer con il mondo esterno si dicono TRASDUTTORI e ne esistono di svariati tipi a seconda delle grandezze in gioco e della potenza

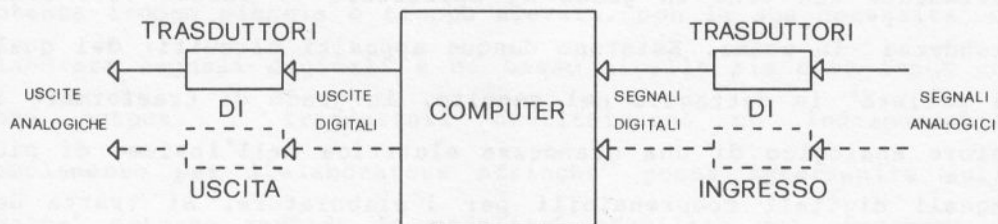



Fig.3 Rilevazione e controllo di grandezze fisiche con un computer

richiesta. I trasduttori di ingresso permettono al computer di rilevare il valore di una grandezza fisica mentre quelli di uscita lo mettono in grado di intervenire effettivamente nei processi ai quali sovrintende. Continuando con l'esempio della temperatura appare subito evidente un altro problema. Infatti il calcolatore puo' interpretare solo segnali di tipo digitale e non analogico quindi la grandezza elettrica proporzionale alla temperatura da controllare ha bisogno di subire un'ulteriore trasformazione, deve cioe' diventare un valore digitale e non piu' analogico. I circuiti elettronici normalmente sono solo in grado di determinare se la tensione o la corrente sono inferiori ad un certo soglia (detta soglia di zero) o superiori ad un'altra (detta soglia di uno) ma certamente non sono in grado di apprezzare il valore vero della grandezza in esame. Esistono dunque appositi circuiti, dei quali si parlera' in dettaglio nel seguito, in grado di trasformare il valore analogico di una grandezza elettrica nell'insieme di piu' segnali digitali comprensibili per l'elaboratore. Si tratta dei convertitori analogico/digitali (A/D) disponibili in una grande varieta' sia dal punto di vista della risoluzione che da quello della precisione e velocita' di conversione. Quando di una grandezza elettrica non interessi il valore vero ma solo se ha superato un certo livello o meno i trasduttori sono concettualmente piu' semplici dovendo solo trasformare un segnale gia' digitale, cioe' a due valori, in segnale elettrico adatto per l'ingresso nell'elaboratore. Tutte queste informazioni che entrano nel computer costituiscono l'Input e sono necessari perche' esso sia in grado di valutare lo stato dell'apparecchiatura sotto controllo.

Discorso analogo si puo' fare nel caso che il computer voglia inviare comandi all'esterno siano essi analogici o digitali. L'uscita evidentemente sara' sempre di tipo digitale (zero o uno)

ed elettricamente costituita da una tensione di basso livello (al massimo qualche Volt) con corrente non superiore a qualche millesimo di ampere. Affinche' questi deboli segnali possano avere effetto su apparecchiature esterne, magari di discreta potenza, sara' necessario quanto meno amplificarli sino al valore richiesto tramite appositi trasduttori di uscita come ad esempio rele' triac o altro. Nel caso che l'uscita debba essere analogica cioe' a piu'di due valori saranno indispensabili circuiti di conversione digitale/analogica (D/A) in grado di compiere l'operazione opposta a quella degli A/D visti prima.

Quanto sopra porta a concludere che il computer ha bisogno di circuiti di interfaccia in grado di adattare le esigenza delle apparecchiature da controllare, cioe' segnali analogici e di potenza troppo piccola o troppo elevata, con la sua necessita' di elaborare segnali digitali e di basso livello sia come input che come output. I trasduttori costituiscono un indispensabile complemento per l'elaboratore affinche' possa intervenire sulla realta' esterna secondo le modalita' stabilite dal programma di lavoro e spesso rappresentano la parte piu' delicata e costosa di un sistema di controllo computerizzato. Si puo' concludere che senza di essi il computer si trova nelle condizioni un cervello intelligente ma senza sensi ne' braccia quindi completamente incapace di utilizzare le sue grandi potenzialita'.

PRIMI ESEMPI DI TRASDUTTORI DI INPUT/OUTPUT

Quanto detto sopra potrebbe aver suggerito l'idea che i trasduttori siano cose strane e mai viste da un utente normale di computer ma questo non e' assolutamente vero. Si pensi dunque al computer stesso, esso e' costituito da circuiti interni dedicati all'elaborazione vera e propria ma ha gia' incorporati diversi trasduttori di input/ouput diversamente non sarebbe possibile

usarlo ne' programmarlo. La tastiera e' il primo di questi circuiti di ingresso perche' permette all'operatore di introdurre informazioni nel computer. Essa, come ognuno puo' rendersi conto, e' costituita da un buon numero di tasti che altro non sono che semplici interruttori a due stati cioe' premuto o non premuto. L'azione dell'operatore su di tasto viene dapprima trasformata in un segnale elettrico digitale cioe' in una tensione superiore od inferiore ai livelli di uno o zero e solo a questo punto l'unita' centrale di elaborazione interna (CPU) e' in grado di ricevere il carattere associato al tasto premuto. Ognuno puo' rendersi conto della natura digitale del segnale prodotto dalla tastiera osservando che in effetti il computer non si accorge di come e quanto profondamente venga premuto un tasto ma solo se si e' superato un certo livello o meno. Un esempio di trasduttore di output noto a tutti e' il video col quale il computer comunica con l'operatore. Un trasduttore piuttosto complesso detto "controllore video" spesso alloggiato all'interno del computer stesso si occupa di tradurre i segnali elettrici digitali (uni e zeri), che costituiscono il messaggio da visualizzare, in segnali analogici in grado di pilotare il fascio di elettroni del tubo catodico per renderlo comprensibile all'operatore. Si puo' dunque osservare che ogni computer, anche molto semplice, deve gia' essere dotato di trasduttori di input e di output tali da permettergli di comunicare con l'esterno altrimenti sarebbe completamente inutilizzabile. Ecco dunque che quando il dialogo debba anche interessare apparecchiature da controllare e' indispensabile adottare appositi circuiti di interfaccia tali da rendere la comunicazione delle informazioni possibile ed esente da errori. La scelta dei tipi di trasduttori e delle modalita' di comunicazione dipendera' evidentemente dal numero, dal tipo e dalle particolarita' delle

grandezze fisiche in gioco in ciascuna specifica applicazione.

TIPI DI NUMERAZIONE

Come gia' accennato una grandezza digitale puo' assumere solo due valori simboleggiati dai numeri 0 e 1. Questo deriva dal fatto che gli elaboratori utilizzano, per i calcoli interni, la numerazione binaria costituita appunto da due soli simboli (0 e 1) a differenza della decimale alla quale si e' abituati che ne utilizza ben 10 (la cifre da 0 a 9).A differenza di come si potrebbe pensare a prima vista, i due tipi di numerazione non differiscono nella sostanza anzi sono molto simili. La numerazione decimale usata quotidianamente e' infatti di tipo posizionale nel senso che ogni cifra assume un valore diverso in funzione della posizione che occupa nel numero. Ad esempio

8363 = 8*1000+3*100+6*10+3*1

o meglio

8363 = 8*10↑3+3*10↑2+6*10↑1+3*10↑0

Ad ogni posizione compete dunque un peso,cioe' un moltiplicatore, pari ad una potenza di 10 . Cosi' la stessa cifra 3 che compare in due posizioni diverse assume un valore ben differente dovuto al diverso moltiplicatore che gli viene applicato in funzione del posto in cui si trova. In tal modo e' possibile rappresentare con solo 10 simboli numeri di qualsiasi grandezza semplicemente utilizzando potenze di 10 sempre piu' elevate e cioe' scrivendo numeri composti da un numero di cifre sempre maggiore senza necessita' di altri simboli oltre i 10 base (le cifre da 0 a 9). Analogamente la numerazione binaria e' posizionale ma utilizza solo due simboli invece dei 10 visti prima per esprimere qualsiasi numero. A esempio

10110 = 1*16+0*8+1*4+1*2+0*1

che tradotto in decimale fa

10110 = 16+4+2 = 22 decimale

la prima espressione si poteva anche scrivere

10110 = 1*2↑4+0*2↑3+1*2↑2+1*2↑1+0*2↑0

Risulta evidente che ad ogni posizione nel numero compete il peso di una diversa potenza di due non piu' di 10. Quindi l'unica vera differenza tra le due numerazioni consiste nel fatto che in quella decimale vi sono 10 simboli mentre nella binaria solo due. Il peso che compete ad ogni posizione naturalmente cresce con le potenze di 10 nel primo caso e con le potenze di 2 nel secondo ma non vi e' alcuna differenza concettuale. Quindi anche con solo due simboli, 0 e 1, e' possibile scrivere qualsiasi numero anche se saranno necessarie molte piu' cifre che non nel caso decimale. Sarebbe a questo punto possibile pensare in modo analogo ad altri tipi di numerazione composte da un numero qualsiasi di simboli ad esempio 5 o 12. Le diverse cifre che comporrebbero i numeri avrebbero peso corrispondente alle potenze di 5 nel primo caso e di 12 nel secondo senza altre differenze dal tipo di numerazione consueto. In effetti oltre la decimale e la binaria in campo informatico hanno acquistato importanza altri due tipi di numerazione: la ottale e la esadecimale. Come dice il nome stesso la prima e' costituita da 8 simboli (i numeri da 0 a 7) e la seconda da sedici. Poiche' in questo secondo caso le cifre della numerazione decimale non sarebbero stati piu' sufficienti oltre il 9 si adottano le prime sei lettere dell'alfabeto e cioe':

DECIMALE	BINARIO	OTTALE	ESADECIMALE
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A

11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

come si puo' osservare i numeri binari sono composti da molte cifre anche per rappresentare valori piccoli a differenza dell'esadecimale che e' il tipo di numerazione piu' compatto. L'ottale era molto usato qualche anno fa mentre oggi e' completamente sostituito dall'esadecimale nella rappresentazione dei numeri contenuti nella memoria interna degli elaboratori per cui conviene esaminarlo un po' piu' in dettaglio.

```
1A3C = 1*16↑3+10*16↑2+3*16↑1+12*16↑0
1A3C = 1*4096+10*256+3*16+12*1
1A3C = 6716 decimale
```

IL BIT

Un termine molto importante che conviene introdurre a questo punto e' il BIT. Esso indica una singola cifra binaria che puo' assumere due soli valori 0 o 1. La parola deriva dalla contrazione del terminee inglese BInary digiT che significa appunto cifra binaria. Ad esempio il numero 1011 e' costituito da 4 bit il primo dei quali e' ad 1, il secondo a zero e cosi' via. Poiche' potrebbero sorgere ambiguita' si suole indicare il bit di peso maggiore come MSB e quello di peso minore con LSB.

Il computer dunque usa la numerazione binaria per le sue elaborazioni interne utilizzando pero' numeri a lunghezza fissa cioe' composti sempre da uno stesso numero di bit al fine di semplificare i suoi circuiti di calcolo. A seconda del tipo di microprocessore adottato le somme avverranno sempre su numeri di 8 bit oppure 16 o 32 cosi' come le differenze e le operazioni logiche di confronto. Il numero di bit che un sistema e' in grado di

elaborare in un sol colpo e' spesso usato per qualificarlo. Questo e' il senso corretto da dare ai termini molto usati tipo "computer a 8 bit" oppure "calcolatore a 16 bit" che vengono frequentemente usate anche a sproposito per indicare caratteristiche diverse di un sistema di elaborazione. Il C64 e' dotato di un microprocessore (unita' centrale di elaborazione) a 8 bit quindi e' in grado di elaborare 8 bit alla volta e non di piu'. Nel caso di somme risulta chiaro che le sue possibilita' si fermano a numeri inferiori al decimale 255 (1111111 binario) ma non c'e' da preoccuparsi perche', se le cifre da trattare sono piu' grandi, esso compira' tante somme successive ad 8 bit quante necessarie senza alcun problema di calcolo ma solo rallentando l'elaborazione. In generale si puo' dire che un'operazione di somma binaria ha una durata che dipende dall'entita' delle cifre in gioco. Anche se le cose spesso non sono cosi' semplici si puo' affermare che un computer a 16 bit e' certamente piu' veloce di uno a 8 nell'esecuzione dei calcoli e quindi nell'elaborazione dlle informazioni. Un numero binario composto da 8 bit e detto BYTE mentre uno a 16, WORD. Quattro bit costituiscono un semibyte o anche un NIBBLE.

LIVELLI ELETTRICI DEI SEGNALI DIGITALI

Un computer e' un complesso circuito elettronico quindi e' ovvio che tutte le informazioni in ingresso ed in uscita dovranno presentarsi sotto forma di grandezze elettriche ed in particolare di tensioni o correnti. Un byte sara' dunque costituito fisicamente da otto linee elettriche, praticamente otto fili, su ciascune delle quali e' presente una tensione associata al livello zero od uno. I valori di tensione associati a ciascuno dei livelli logici possono differire da un computer ad un altro, in alcuni casi -12 Volt significa zero e +12 uno oppure viceversa, l'essenziale e' che non

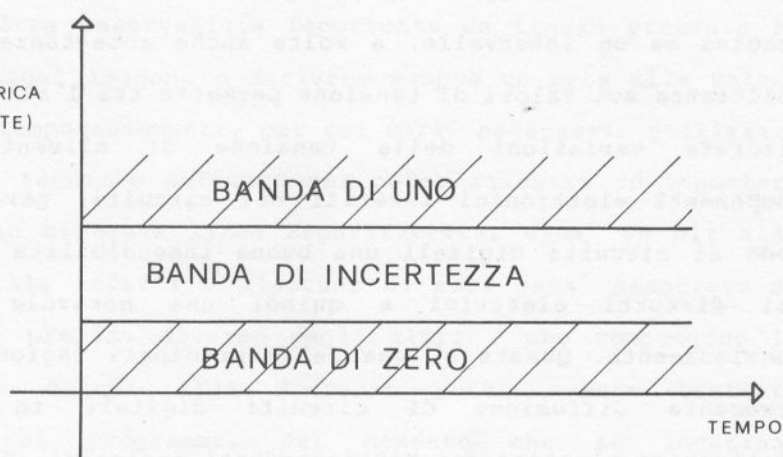



Fig.4 Intervalli di riconoscimento del valore logico (0-1) associato ad una grandezza elettrica

si possano creare ambiguita' nel qual caso l'informazione verrebbe recepita in modo non corretto. I computer tipo C64 usano i livelli logici TTL (Transistor-Transistor Logic) come valori logici ed e' la convenzione attualmente piu' diffusa. Questo significa che uno zero logico e' rappresentato da una tensione compresa tra 0 e 0,9 Volt mentre un uno da valori tra 2,4 a 5 Volt. Come si vede si avra' ambiguita', cioe' non sara' possibile prevedere come verranno interpretate le cifre, per tensioni comprese tra 0,9 e 2,4 Volt che quindi sono assolutamente da evitare. Un aspetto da tenere presente e' che ad ogni valore logico non e' associata una tensione precisa ma un intervallo, a volte anche abbastanza ampio. Questa tolleranza sui valori di tensione permette tra l'altro di accettare discrete variazioni della tensione di alimentazione e dei componenti elettronici inseriti nei circuiti, garantendo in tal modo ai circuiti digitali una buona insensibilita' nei confronti dei disturbi elettrici e quindi una notevole sicurezza di funzionamento. Questa e' una delle pricipali ragioni della sempre crescente diffusione di circuiti digitali in ogni settore elettronico al posto dei corrispondenti analogici. Basti pensare al campo della riproduzione musicale ad alta fedelta' che sta trasformandosi sempre piu' da analogica a digitale ed all' introduzione del "video-disco. I circuiti digitali sono normalmente piu' complessi dei corrispondenti analogici ma hanno il grande vantaggio di essere molto piu' immuni da disturbi ed insensibili alle variazioni dei valori dei componenti elettronici e normalmente non necessitano di sofisticate operazioni di taratura e messa a punto per svolgere le loro funzioni.

SOFTWARE PER LA GESTIONE DELLE LINEE DI I/O

Prima di esaminare dettagliatamente i circuiti che permetteranno

al C64 di leggere e scrivere lo stato delle linee di comando esterne conviene affrontare i problemi relativi alla gestione di valori binari con il linguaggio Basic che, come noto, utilizza solo la numerazione decimale per l'ingresso e la visualizzazione dei dati. Occorre a questo punto premettere che le linee di I/O dedicate al dialogo con l'esterno saranno viste dal programmatore come normali locazioni di memoria ad indirizzi particolari e potranno tranquillamente essere gestite utilizzando le istruzioni del linguaggio Basic per l'accesso diretto alla memoria e cioe' POKE e PEEK. Altra osservazione importante da tenere presente e' che tali istruzioni leggono o scrivono sempre un byte alla volta, cioe' 8 bit contemporaneamente, per cui sara' necessario utilizzare alcune semplici tecniche software per poter rilevare od impostare al valore voluto ciascuna linea separatamente, cioe' un bit alla volta. In generale infatti a ciascuno di essi sara' associato un significato ben preciso diverso degli altri 7 che compongono lo stesso byte e quindi essi dovranno poter essere trattati separatamente nel programma. Dal momento che le locazioni corrispondenti alle linee di I/O non differiranno in alcun modo dalle normali celle di memoria si potra' fare pratica anche prima di avere realizzato l'espansione di input/output prendendo una qualunque di queste ultime purche' non utilizzata dal Basic onde non incorrere in spiacevoli inconvenienti o in strani malfunzionamenti del sistema. Una di esse puo' essere la locazione 251 (esadecimale FB) e verra' usata nel seguito come finta porta di I/O. Con le istruzioni

```
10 A%=PEEK(251)
20 PRINT A%
99 END
```

il valore binario contenuto nella locazione 251 verra' trasferito

nella variabile A% e quindi convertito in decimale per la visualizzazione. Derivando da un numero binario a otto bit il suo valore sara' compreso tra 0 e 255. Viceversa con

```
100 B%=123
110 POKE251,B%
199 END
```

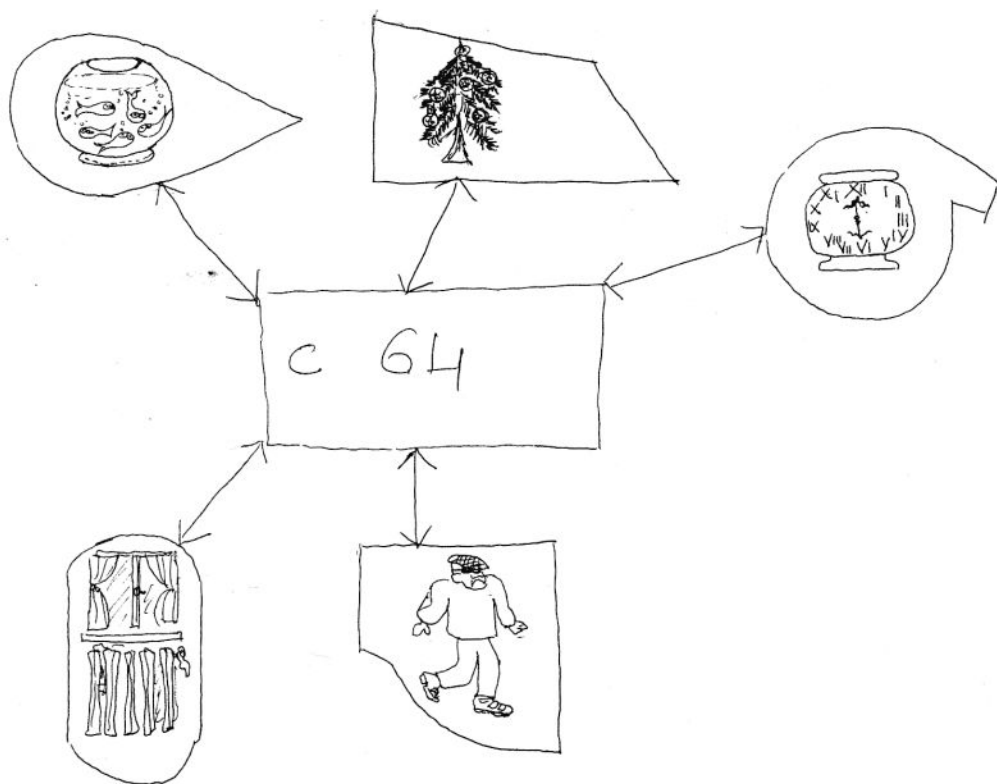
il valore decimale caricato in B% (nell'esempio 123) verra' convertito in binario e scritto nell'indirizzo 251. Provando ad eseguire la prima volta con RUN 10 comparira' sul video un numero qualsiasi, sempre pero' compreso tra 0 e 256, corrispondente al contenuto della locazione letta al momento dell'accensione del sistema. Eseguendo poi RUN 100 e di seguito RUN 10 la risposta sara' 123, cio' significa che la cella di memoria scelta (251) contiene ora il valore voluto (123). Provando con diversi valori di B% e' possibile impostare al valore voluto la locazione scelta. Immaginando pero' ciascuno degli otto bit del byte 251 come una linea di I/O conviene ora cercare di visualizzarne il valore singolarmente che naturalmente potra' essere solo zero od uno. Analogamente come secondo passo si dovra' cercare di impostare il valore voluto, sempre zero od uno, nel bit voluto senza turbare il valore dei rimanenti.

I programmi Prg 1 e Prg 2 rappresentano un esempio di come si possano gestire singolarmente i bit di una locazione di memoria e di conseguenza le linee di ingresso o uscita del sistema.

```
1 rem***lettura dalla locazione 251
2 rem***e scomposizione nei bit singoli
8 print
10 a%=peek(251):rem lettura locazione in a%
15 x%=128:rem massima potenza di 2 in 8 bit
20 fori=7to0step-1:rem ripete per 8 bit
30 b%(i)=int(a%/x%):a%=a%-b%(i)*x%:rem calcola il valore del bit
40 x%=x%/2:rem potenza di 2 inferiore
43 next
45 print"leggo da  251   b7 b6 b5 b4 b3 b2 b1 b0"
46 print"                ";
50 fori=7to0step-1:printb%(i);:next
60 print:print
80 print"scrivo su 251   b7 b6 b5 b4 b3 b2 b1 b0"
90 print"                ";
93 rem***input del valore dei bit singoli
95 rem***e preparazione del valore per poke
100 fori=7to0step-1
110 getc$:ifc$=""then110
115 c%(i)=val(c$)
120 ifc%(i)<0orc%(i)>1then110:rem solo zeri e uni
130 printc%(i);:next
133 print:print
135 x%=128:d%=0:rem preparazione variabili
140 fori=7to0step-1:rem ripete per 8 bit
150 d%=d%+c%(i)*x%:x%=x%/2:rem calcolo del valore
160 next
170 poke251,d%:rem scrittura in 251
180 goto10:rem torna a leggere
```

PRG1:visualizzazione e scrittura di una
locazione di memoria a bit singoli

```
1 rem lettura e scrittura di singoli
2 rem bit in una locazione di memoria
3 rem
10 x=251:rem indirizzo locazione intersessata
15 print"shift+clr/home":print"bit                 7 6 5 4 3 2 1 0"
20 print"bit numero ";: rem nella variabile
30 getn$:ifn$=""then30 : rem n il numero di bit (0-7)
40 n=val(n$):ifn>7then30 :interessato
50 printn:poke(1098-n*2),(peek(1098-n*2)+128)
60 print"(l)eggi o (s)crivi" :rem scelta tra
70 getls$:ifls$=""then70      :rem lettura o
80 ifls$="l"then200            :rem scrittura
90 ifls$<>"s"then70
97 rem
98 rem scrittura di un bit
99 rem
100 print"zero 0/1 uno";       :rem scelta del
110 getv$:ifv$=""then110       :rem valore
120 v=val(v$):ifv>1orv<0then110:rem tra zero e uno
130 printv:poke(1138-n*2),v+48
145 rem****istruzione che pone in x il valore v (0/1)
146 rem****nel bit di numero n (0-7)
150 pokex,((peek(x)and(255-2↑n))+v*2↑n)
160 rem
170 goto230
197 rem
198 rem lettura di un bit
199 rem
200 rem*****istruzione che legge il valore (0/1)
203 rem*****in v del bit n della locazione x
205 v=((peek(x)and(2↑n))/2↑n)
206 rem
210 poke(1138-n*2),v+48
220 getls$:ifls$=""then220
230 goto10
```

PRG2:set o reset di bit singoli
in una locazione di memoria

SCHEDA DI ESPANSIONE DELL'INPUT/OUTPUT

PER COMMODORE C64

In questo capitolo verra' affrontato il problema di come introdurre informazioni digitali nel C64 o come fare in modo che esso possa comunicarle ad un'altra apparecchiatura. Per ottenere cio' e' necessario aggiungere al computer stesso una scheda di espansione in modo da avere a disposizione le linee fisiche per mezzo delle quali comunicare i segnali digitali ad altri circuiti

elettronici. Il C64 in effetti e´ gia munito di 10 linee di questo tipo disponibili nella cosiddetta "USER PORT" ma esse vengono gia´ utilizzate dal computer per altre funzioni, ad esempio la comunicazione seriale, quindi la cosa migliore e´ realizzare un input/output separato al fine di non dover interferire in alcun modo con i software interno del computer o sacrificarne importanti funzioni.

IL CONNETTORE DI ESPANSIONE

Per BUS si intende un insieme di linee sulla quale vengono scambiate informazioni digitali tra circuiti elettronici. Il C64 e´ dotato nella parte posteriore di un connettore a 22+22 linee sul quale sono disponibili tutti i segnali necessari all´espansione dell´hardware del computer stesso. Con riferimento alla configurazione delle linee di espansione esse possono essere distinte in diversi gruppi. L´alimentazione per i circuiti esterni puo´ essere prelevata dai pin 22, 1, Z e A come massa e dai 2 e 3 come tensione positiva a 5 Volt purche´ l´assorbimento non sia superiore a qualche centinaio di milliampere. I pin dal 21 al 14, indicati come CD0-CD7, costituiscono il bus dati, cioe´ le linee per mezzo delle quali il microprocessore comunica con tutti gli altri componenti del circuito. Come si puo´ vedere esso e´ costituito da 8 linee con CD0 LSB e CD7 MSB essendo la CPU 6510 a otto bit. Questo bus e´ molto importante e viene usato sia come ingresso che come uscita per i dati a seconda del livello presente sul pin 5 CR/W. Un uno significa dato entrante ed uno zero dato uscente dalla CPU. IL pin C (RESET) si trova ad uno durante il normale funzionamento ed a zero durante la fase di accensione del C64. Sul pin E (SO2) si trova un´onda quadra di frequenza pari ad 1 Mhz che serve come segnale di sincronismo per tutti i componenti

CONTATTI DEL CONNETTORE DI ESPANSIONE DEL C64

CONTATTI SUPERIORI

PIN	NOME	FUNZIONE
1	GND	negativo dell'alimentazione
2	+5V	5 Volt alimentazione comune
3	+5V	come sopra
4	IRQ	input per interruzioni non mascherabili
5	CR/W	linea di lettura/scrittura di unita' esterne (Scrittura = 0)
6	Dot clock	frequenza punti sul video
7	I/O1	linea di aspansione 1 per indirizzi tra DE00 a DEFF dec. 56832-57087
8	GAME	ingresso per segnalare una Cardrige esterna
9	EXROM	ingresso per segnalare una ROM esterna
10	I/O2	linea di espansione 2 per indirizzi tra DF00 a DFFF dec. 57088-57343
11	ROML	linea di espansione per ROM esterna tra 8000 a 9FFF dec. 32768-40959
12	BA	Bus disponibile per accesso diretto a memoria (DMA)
13	DMA	accesso diretto alla memoria
14	CD7	bus dati bit 7 (MSB)
15	CD6	" " " 6
16	CD5	" " " 5
17	CD4	" " " 4
18	CD3	" " " 3
19	CD2	" " " 2
20	CD1	" " " 1
21	CD0	" " " 0 (LSB)
22	GND	negativo alimentazione

CONTATTI INFERIORI

PIN	NOME	FUNZIONE
A	GND	negativo alimentazione
B	ROMH	Rom esterna tra A000-BFFF dec.40960-49151 oppure tra E000-FFFF dec.57344-65535
C	RESET	Reset di sistema
D	NMI	ingresso per interruzioni non mascherabili
E	SO2	Clock di sistema a 1 Mhz (fase 2)
F	CA15	bus indirizzi bit 15 (MSB)
H	CA14	" " " 14
J	CA13	" " " 13
K	CA12	" " " 12
L	CA11	" " " 11
M	CA10	" " " 10
N	CA9	" " " 9
P	CA8	" " " 8
R	CA7	" " " 7
S	CA6	" " " 6
T	CA5	" " " 5
U	CA4	" " " 4
V	CA3	" " " 3
W	CA2	" " " 2
X	CA1	" " " 1
Y	CA0	" " " 0 (LSB)
Z	GND	negativo alimentazione

del sistema. Altro bus molto importante e' quello costituito dalle linee CA0-CA15 (pin Y-H) detto bus degli indirizzi perche' su di esso la CPU specifica l'indirizzo della locazione di memoria sulla quale intende operando. Si tratta di 16 linee quindi gli indirizzi possibili sono pari ai mumeri binari esprimibili con 16 bit e cioe' 65536 o come si dice 64 Kbyte dal momento che 1 Kbyte corrisponde a 1024 Byte.Un'altra linea molto importante e' chiamata R/W e serve alla CPU per segnalare se il dato presente sul bus dati e' uscente (R/W = 0) o entrante (R/W = 1) e cio' e' indispensabile per coordinare il trasferimento delle informazioni tra i componenti del sistema e l' unita' centrale di elaborazione. Oltre a queste, che rappresentano i principali segnali di un tipico sistema a microprocessore, sono presenti sul bus di espansione altre linee, tipiche del C64, utili per facilitare la connessione di dispositivi esterni al computer. Si tratta dei segnali BLK1, BLK2, BLK3, BLK5, RAM1, RAM2, RAM3, IO1 e IO2 che si trovano normalmente al valore logico 1 e scendono a zero quando il programma intende accedere a particolari aree di memoria. Molto utili per il nostro scopo si presentano le ultime due, cioe' IO1 e IO2, che segnalano, scendendo a zero, la lettura o la scrittura di una cella di memoria tra 56832 (esadecimale DE00) e 57087 (DEFF) per IO1 e tra 57088 (DF00) e 57343 (DFFF). In questi indirizzi, inutilizzati duranteil normale lavoro del Basic, verranno allocate le porte di input/output per espandere il C64 che permetteranno, senza interferire con il normale funzionamento del computer, di comunicare con apparecchiature esterne. Come cio' si possa realizzare verra' esaminato in dettaglio in seguito.

IL PIA (PERIPHERAL INTERFACE ADAPTER)

Per facilitare le operazioni di comunicazione con il mondo esterno

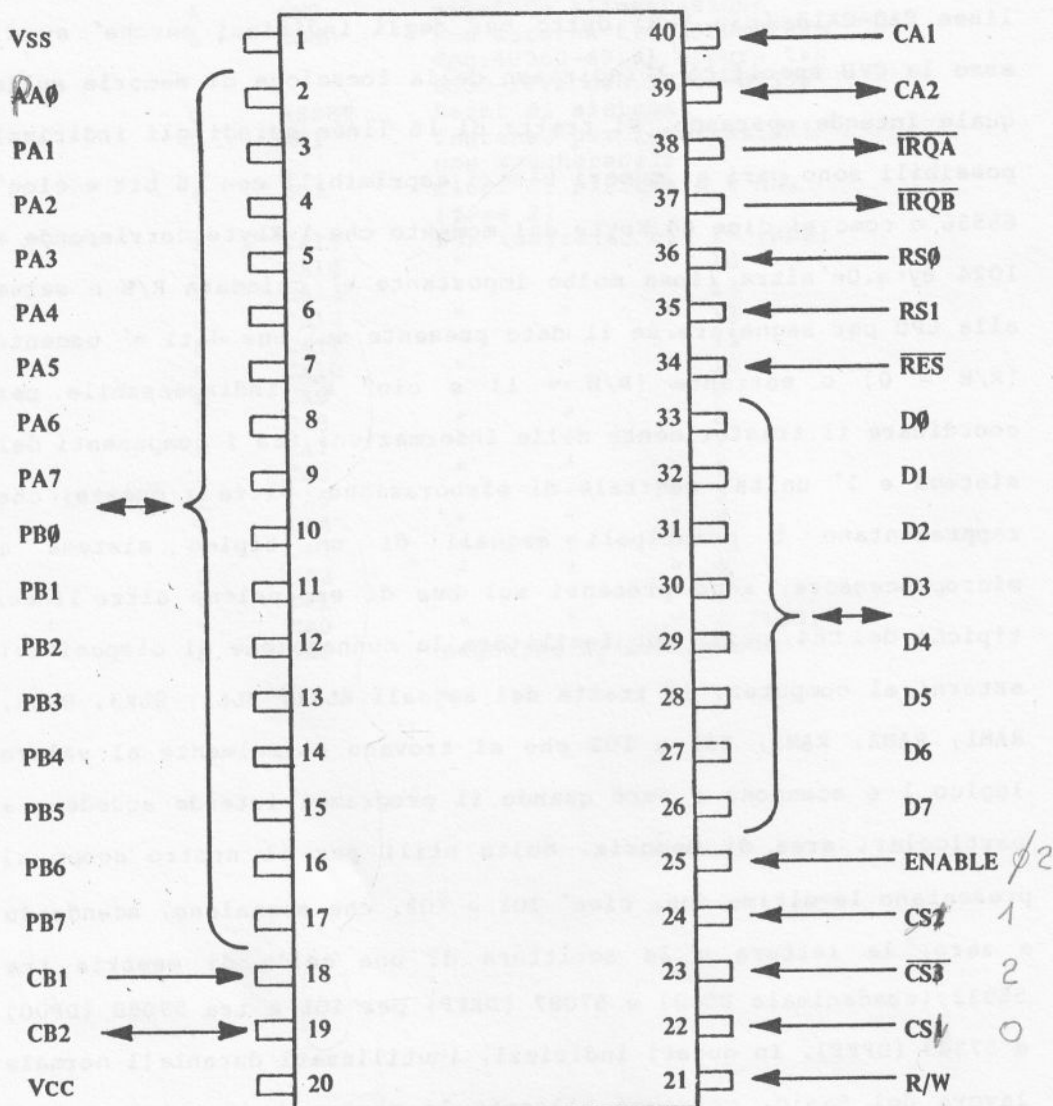



Fig.5 Piedinatura del chip 6521 PIA adatto per espandere le linee di ingresso/uscita del C64

uno dei circuiti piu' adatti e' rappresentato dal chip di I/O R6521 denominato PIA cioe' adattatore per interfacciamento di periferiche. Esso e' stato introdotto sul mercato da diversi anni per facilitare il collegmento di periferiche con tastiere, stampanti, display ed altro con i microprocessori delle famiglie 6502 e 6800 Motorola. La sua funzione specifica e' quella di adattare i segnali provenienti da unita' elettroniche od elettromeccaniche esterne con i Bus interni di un sistema a microprocessore. Da una parte il 6521 si collega alle periferiche tramite due gruppi di otto linee detti PORT A e PORT B con le relative linee di controllo CA1 e CA2 per il PORT A e CB1 e CB2 per il PORT B. Dall'altra parte esso dialoga direttamente con il microprocessore tramite un Bus dati ad otto bit e le relative linee di controllo gia' esaminate per il Bus di espansione del C64. Lo schema funzionale riesce a chiarire la struttura interna di questo chip di interfaccia. Inoltre questo dispositivo e' programmabile in molte sue funzioni direttamente via software durante l'inizializzazione al fine di adattarsi di volta in volta alle caratteristiche del dispositivo da interfacciare. Questo e' effettivamente molto importante perche' permette di disporre di una stessa interfaccia anche quando i dispositivi da collegare al computer siano diversi di volta in volta. In fig. e' schematizzata la struttura interna del PIA in modo da chiarirne i registri interni, le loro connessioni e le diverse porte di I/O del dispositivo. Nei paragrafi seguenti verranno discusse le linee di ingresso e di uscita del PIA insieme alle funzioni di ciascun registro. Infine il 6521 verra' visto dal punto di vista applicativo descrivendo le modalita' di gestione delle diverse linee dei PORT in funzione dei diversi tipi di periferiche interfacciabili.

LA PROGRAMMAZIONE DEL PIA

Il 6521 puo′ essere pensato come costituito da due sezioni distinte denominate "parte A" e "parte B", ciascuna delle quali e′ composta da un registro di controllo CRA o CRB, da un registro di direzione DDRA o DDRB e da un registro di output ORA o ORB. Il dispositivo e′ dotato anche di particolari circuiti in grado di provocare interruzioni (INTERRUPT) al microprocessore al verificarsi di determinati eventi esterni. Questo segnale di interruzione costringe la CPU ad interrompere lo svolgimento normale del programma in corso per eseguire determinate funzioni richieste dall′evento esterno e solo alla loro conclusione essa potra′ riprendera′ il lavoro interrotto. Poiche′ una gestione di questo tipo rischia di interferire pesantemente con il funzionamento normale del C64 si ritiene opportuno, per il momento, di non ricorrervi, quindi gli aspetti suddetti verranno solamente accennati nella presente trattazione. I registri interni del 6521 sono dunque, a partire dall′indirizzo base del dispositivo:

BASE+0	DDRA	ORA
BASE+1	CRA	
BASE+2	DDRB	ORB
BASE+3	CRB	

Come si puo′ osservare si hanno sei registri in sole quattro locazioni di memoria. CRA e CRB sono sempre disponibili mentre se il bit 3 del rispettivo registro di controllo e′ ad uno saranno disponibili i registri di ingresso/uscita mentre diversamente sara′ possibile accedere ai registri di direzione di ciascuno dei PORT. Il bit 3 dunque di ciscun registro di controllo CRA e CRB serve per selezionare alle locazioni BASE+0 e BASE+2 i registri di direzione o quelli di output. I registri di output servono per comunicare effettivamente con l′esterno mentre quelli di direzione definiscono se ciscuna linea dei PORT debba agire da ingresso o da uscita.

CONNESSIONI DEL CHIP DI I/O
6521 PIA

PIN	NOME	DIR	FUNZIONE
1	Vss		negativo alimentazione
2	PA0	I/O	Port A linea 0 (LSB)
3	PA1	I/O	" A " 1
4	PA2	I/O	" A " 2
5	PA3	I/O	" A " 3
6	PA4	I/O	" A " 4
7	PA5	I/O	" A " 5
8	PA6	I/O	" A " 6
9	PA7	I/O	" A " 7 (MSB)
10	PB0	I/O	Port B linea 0 (LSB)
11	PB1	I/O	" B " 1
12	PB2	I/O	" B " 2
13	PB3	I/O	" B " 3
14	PB4	I/O	" B " 4
15	PB5	I/O	" B " 5
16	PB6	I/O	" B " 6
17	PB7	I/O	" B " 7 (MSB)
18	CB1	I	linea di controllo 1 del port B
19	CB2	I/O	linea di controllo 2 del port B
20	Vcc		alimentazione positiva +5 Volt
21	R/W	I	lettura/scrittura reg. collegare a CR/W
22	CS1	I	linea di abilitazione 1 attiva ad 1
23	CS3	I	linea di abilitazione 3 attiva a 0
24	CS2	I	linea di abilitazione 2 attiva ad 1
25	ENABLE	I	ingresso di sincronismo collegare a SO2
26	D7	I/O	Bus dati bit 7 (MSB) collegare a CD7
27	D6	I/O	Bus dati bit 6 coll.CD6
28	D5	I/O	" " " 5 " CD5
29	D4	I/O	" " " 4 " CD4
30	D3	I/O	" " " 3 " CD3
31	D2	I/O	" " " 2 " CD2
32	D1	I/O	" " " 1 " CD1
33	D0	I/O	" " " 0 (LSB) collegare a CD0
34	RES	I	ingresso per reset collegare con RESET
35	RS1	I	selezione registri 1 collegare a CA1
36	RS0	I	selezione registri 0 collegare a CA0
37	IRQB	O	uscita per interruzioni non mascherabili dal

			Port B,coll. a IRQ
38	IRQA	O	uscita per interruzioni non mascherabili dal Port A,coll. a IRQ
39	CA2	I/O	linea di controllo 2 del Port A
40	CA1	I	linea di controllo 1 del Port A

```
10 rem ****esempio di programmazione***
11 bl=2:   rem  bit 1
12 b2=4:   rem  bit 2
13 b3=8:   rem  bit 3
14 b4=16:  rem  bit 4
15 b5=32:  rem  bit 5
16 b6=64:  rem  bit 6
17 b7=128: rem  bit 7
19 ren ****esempio di programmazione****
20 rem ****pa7 e pal in output ********
30 rem ****le rimanenti linee input****
40 la=b7+bl:lb=0:gosub9000
50 end:rem linee programmate !*********
9000 rem*****inizializzazione pia*******
9010 rem*****in la definizione port a **
9020 rem*****in lb definizione port b **
9025 rem
9028 rem******indirizzo base df84 ******
9030 p=13*16↑3+15*16↑2+8*16+4
9040 pa=p  :rem indirizzo port a (ddra e ora)
9050 pb=p+2:rem indirizzo port b (ddrb e orb)
9060 da=p+1:rem registro di controllo a
9070 db=p+3:rem registro di controllo b
9100 rem******inizilizzazione porte*****
9105 rem
9107 rem scopre il registro di direzione a
9110 pokep+1,0
9115 rem
9117 rem scopre il registro di direzione b
9120 pokep+3,0
9125 rem
9127 rem programma le linee del port a (0=inp 1=out)
9130 pokep,la
9135 rem
9137 rem programma le linee del port b (0=inp 1=out)
9140 pokep+2,lb
9145 rem
9147 rem scopre il registro a vero e proprio
9150 pokep+1,4
9155 rem
9157 rem scopre il registro b vero e proprio
9160 pokep+3,4
9170 rem******fine inizializzazione****
9180 return
```

PRG 3. Routine per la programmazione dei Port del PIA 6521

Scrivendo uno nel rispettivo bit si fa agire la corrispondente linea da uscita mentre con zero essa diventa un ingresso. Ad esempio se, avendo posto a zero il bit 3 del CRA si scrive in BASE+0 il valore decimale 240 (esadecimale $F0 e binario 11110000) si comunica al PIA che le quattro linee di alto ordine del PORT A (da PA4 a PA7) devono agire da output mentre quelle di basso ordine (da PA0 a PA3) da input. Qualunque combinazione di uni e di zeri e' ammessa e la definizione delle linee puo' essere anche modificata piu' volte, se necessario, durante l'esecuzione di un programma. Naturalmente per leggere i dati esterni o settare le linee di uscita sara' indispensabile prima porre ad uno il bit 3 del CRA per rendere accessibili i registri di input e di output del dispositivo. Nello stesso modo si possono programmare le linee del PORT B agendo naturalmente sul registro CRB (BASE+3) e DDRB o ORB (BASE+2). Scrivendo un uno nel bit di ORA od ORB relativo ad una linea programmata come output (il bit corrispondente del registro di direzione e' ad uno) si portera' la linea esterna ad una tensione superiore a 2.7 Volt, mentre con zero essa si portera'circa a 0 Volt. Scrivendo nel bit orrispondente ad una linea programmata come ingresso non si ha provoca alcun effetto all'esterno del dispositivo. Viceversa se una linea e stata definita di input il relativo bit sara' ad uno se la tensione esterna sul pin corrispondente supera la soglia di 2.7 V o zero nel caso opposto. In tal modo e' possibile gestire le linee esterne come i bit di una normale locazione di memoria del C64.

LA SCHEDA DI ESPANSIONE DI INPUT OUTPUT

Per poter disporre di linee con le quali comunicare con altri dispositivi senza pero' interferire con le normali funzioni del C64

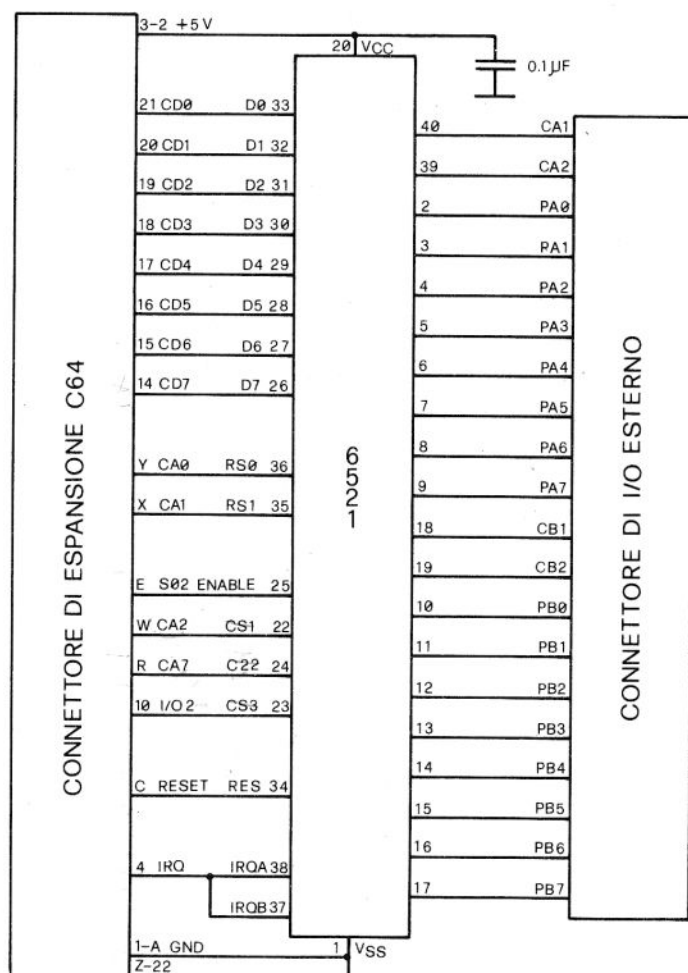



Fig.6 Schema elettrico della scheda di espansione di input/output per il C64

e' dunque necessario collegarvi uno o piu' chip periferici (PIA) a seconda delle necessita'. In Fig. sono indicate le connessioni da eseguire tra il Bus di espansione del computer ed il 6521 affinche' esso possa essere letto e scritto dal microprocessore come normali locazioni di memoria RAM. L'unico aspetto che vale la pena di approfondire riguarda la definizione dell'indirizzo base relativo a ciascun PIA connesso con il Bus in modo che non possano sorgere ambiguita'. Infatti e' indispensabile che durante l'esecuzione di un programma si possano leggere o scrivere i registri di ciascuno dei chip di I/O senza interferire con gli altri o peggio con altri componenti del sistema. Un PIA viene abilitato solo quando si trovano a livello uno contemporaneamente i pin 22 e 24 (CS0 e CS1) ed a zero il pin 23 (CS2) come si puo' desumere dalle note applicative. Collegando dunque al pin 23 la linea IO2 del Bus di espansione e fissando ad uno i pin 22 e 24 (collegandoli a + Vcc) il PIA verra' attivato ogni volta che si accede ad un indirizzo che manda a zero la linea IO2 e cioe' compreso tra 57088 e 57343 ($DFOO $DFFF). In tutte le 256 locazioni di questo spazio di memoria si ripeteranno le quattro locazioni caratteristiche dei registri interni del PIA. Se non e' necessario avere altri dispositivi di I/O collegati sul Bus la soluzione e' certamente accettabile ma non sempre puo' bastare. Infatti collegando ad esempio il pin 22 (CS0) alla terza linea del Bus degli indirizzi (CA2) ed il 24 alla ottava (CA7) per attivare il PIA ora non e' piu' sufficiente che l'indirizzo sia nell'intervallo precedente (da $DFOO a $DFFF) ma contemporaneamente anche CA2 e CA7 dovranno trovarsi al livello logico uno. Solo a partire dagli indirizzi che soddisfano tutte e tre le condizioni precedenti saranno accessibili i registri del PIA, diversamente esso non verra' minimamente interessato dalle operazioni del

sistema. In questo caso di ottiene un indirizzo base pari a 57220 ($DF84) oppure 57236 ($DF94) o altri comunque compresi tra 57088 ($DF00) e 57343 ($DFFF), che rappresentano l'area di memoria nella quale IO2 va a zero, e dalla presenza contemporanea di un uno in CA2 e CA7. Naturalmente collegando altre linee del Bus degli indirizzi ai pin 22 e 24 di diversi PIA essi potranno coabitare, senza interferenze, nella stessa pagina di memoria e venire abilitati singolarmente secondo le esigenze di programmazione. Si dovra' porre pero' molta attenzione perche' vi saranno sempre molti indirizzi, compresi nella pagina vista, in grado di attivare lo stesso dispositivo ed inoltre piu' PIA potranno essere selezionati contemporaneamente da diversi altri. Sara' cura del programmatore individuare ed evitare accuratamente la lettura o la scrittura di questi ultimi al fine di non creare spiacevoli conflitti tra i diversi chip di I/O di espansione. Come si puo' facilmente immaginare anche i dispositivi interni al C64 od a qualunque altro computer sono indirizzati in modo simile e da cio' si comprende come gli strani numeri che si devono usare nelle istruzioni POKE per attivare particolari funzioni siano stati stabiliti da parte dei progettisti hardware. I semplici programmi di esempio permetteranno di chiarire ulteriormente il procedimento da seguire per programmare in input od in output le linee dei due port disponibili per ogni PIA.

LE LINEE DI CONTROLLO DEI PORT

Le linee CA1 e CA2 del PORT A insieme a CB1 e CB2 del PORT B costituiscono un caso particolare di ingressi/uscite per il PIA. Infatti mentre leggendo le locazioni corrispondenti ai PORT di I/O del 6521 si determina se ciascuna linea si trova al valore uno o a zero, nel caso delle linee di controllo cio' non e' possibile

poiche' esse non sono sensibili ai livelli logici esterni ma alle loro variazioni. Nel bit di piu' alto ordine (IRQA1) di CRA si trovera' il valore uno se sul pin esterno collegato a CA1 e' avvenuto un cambio di livello secondo le modalita' specificate con i due bit di basso ordine dello stesso registro. Analogamente per le altre linee di controllo i valori letti vanno interpretati secondo la tabella indicata. Il valore letto nei bit associati a questi particolari ingressi non rispecchiano quindi il livello logico esterno ma piuttosto se esso ha subito una variazione. Cio' puo' essere molto utile nel caso di segnali variabili in modo molto rapido per i quali c'e' rischio di non riuscire a catturare il breve istante di variazione. IRQA1 invece manterra' inalterata l'informazione del cambio di livello anche se prima della lettura la linea esterna sara' ritornata al livello precedente e quindi apparentemente nulla e' avvenuto. Si dice dunque che le linee dei port sono sensibili al livello dei pin esterni a loro associati (LEVEL SENSITIVE) mentre quelle di controllo alle variazioni (EDGE SENSITIVE). Leggendo i bit relativi alle linee di controllo e' dunque possibile stabilire se durante l'intervallo di tempo intercorso tra la lettura precedente e l'attuale lo stato logico e' rimasto sempre stabile oppure se, anche per un breve istante, e' cambiato ritornando magari di nuovo al livello originario. La segnalazione della avvenuta transizione sulle linee di controllo viene cancellata ogni volta che si legge o si scrive il port corrispondente, cioe' il PORT A per CA1 e CA2 ed il PORT B per CB1 e CB2. Attenzione dunque perche' operando sui port si possono inavvertitamente cancellare le informazioni realtive ai bit di controllo che quindi dovranno essere preventivamente salvate se necessarie alla gestione in corso. Inoltre a differenza delle linee CB1 e CA1 che possono solo fungere da ingressi, per CA2 e CB2 e'

STRUTTURA DI CRA (BASE+1)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
IRQA1	IRQA2	PROGRAMM. DI CA2			ORA/ DDRA	PROGRAMM. CA1	

STRUTTURA DI CRB (BASE+3)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
IRQB1	IRQB2	PROGRAMM. DI CB2			ORB/ DDRB	PROGRAMM. CB1	

PROGRAMMAZIONE DI CA1 (CB1) IN CRA (CRB)

bit 1	bit 0	tipo di transizione
0	0	da alto a basso
1	0	da alto a basso

PROGRAMMAZIONE IN INPUT DI CA2 (CB2) IN CRA (CRB)

bit 5	bit 4	bit 3	tipo di transizione
0	0	0	da alto a basso
0	1	1	da basso a alto

PROGRAMMAZIONE IN OUTPUT DI CA2 (CB2) in CRA (CRB)

bit 5	bit 4	bit 3	stato di uscita
1	1	0	livello basso (zero)
1	1	1	livello alto (uno)

```
1 rem *****definizione bit ************
10 b0=1:   rem  bit 0
11 b1=2:   rem  bit 1
12 b2=4:   rem  bit 2
13 b3=8:   rem  bit 3
14 b4=16:  rem  bit 4
15 b5=32:  rem  bit 5
16 b6=64:  rem  bit 6
17 b7=128: rem  bit 7
19 ren ****esempio di programmazione****
20 rem **** per ca2 e cb2 *************
30 rem
40 ca=0:gosub9500:rem ca2 a zero
50 cb=1:gosub9600:rem cb2 a uno
60 end
9490 rem
9500 rem****routine di programmazione**
9510 rem****        linea ca2         **
9520 rem****ca contiene il valore     **
9550 rem******indirizzo base df84 ******
9560 p=13*16↑3+15*16↑2+8*16+4
9570 pr=b5+b4:rem 00110000 :rem ca2 a zero
9580 ifca=othen9590:rem se ca=0 ok
9585 pr=pr+b3:rem 00111000 :rem ca2 a uno
9590 poke(p+1),(peek(p+1)and(255-b5-b4-b3)or pr)
9595 return
9599 rem
9600 rem****routine di programmazione**
9610 rem****        linea cb2         **
9620 rem****cb contiene il valore     **
9650 rem******indirizzo base df84 ******
9660 p=13*16↑3+15*16↑2+8*16+4
9670 pr=b5+b4:rem 00110000 :rem cb2 a zero
9680 ifcb=othen9690:rem se cb=0 ok
9685 pr=pr+b3:rem 00111000 :rem cb2 a uno
9690 poke(p+3),(peek(p+3)and(255-b5-b4-b3)or pr)
9695 return
```

PRG4. Routine per la programmazione
in output delle linee CA2 e CB2
del PIA

possibile anche programmarle come output. Con riferimento alla tabella relativa ai bit di CRA e CRB e' possibile utilizzare anche queste due ultime linee come uscite in caso di necessita', anche se la loro gestione e' un po' piu' macchinosa di quella delle normali linee dei port. In conclusione un PIA permette al sistema di avere a disposizione ben 20 linee di input/output programmabili singolarmente per l'interfacciamento di unita' esterne. Come si e' visto e' agevole espandere ulteriormente l'interfaccia di I/O semplicemente aggiungendo altri chip di tipo 6521, in modo da soddisfare qualsiasi esigenza di controllo basata sul C64.

PRIMI ESEMPI D'USO DELLE LINEE DI I/O

LA SCHEDA DI VALUTAZIONE

Dopo aver costruito e collegato l'interfaccia al C64 e' necessario scrivere alcuni semplici programmi per acquistare famigliarita' con il suo uso e la sua programmazione. A tal fine e' molto utile realizzare una schedina in grado di simulare la presenza di unita' esterne per iniziare a gestirne i segnali sia di ingresso che di uscita. Con otto diodi led e otto interruttori unipolari e' possibile molto semplicemente sia introdurre dati nel computer che visualizzare lo stato delle linee di output. Riguardo al semplice schema proposto e' opportuno fare alcune osservazioni dal punto di vista elettrico. Infatti, come si puo' notare, chiudendo un interruttore si obbliga la linea del PIA a portarsi al livello logico zero mentre lasciandolo aperto essa risulta aperta cioe' non collegata a nulla. Questo non costituisce un problema nel nostro caso in quanto una linea di ingresso non collegata a nulla rileva sempre un livello logico uno, ma in caso di applicazioni destinate ad operare in ambienti ricchi di disturbi di natura elettrica cio'

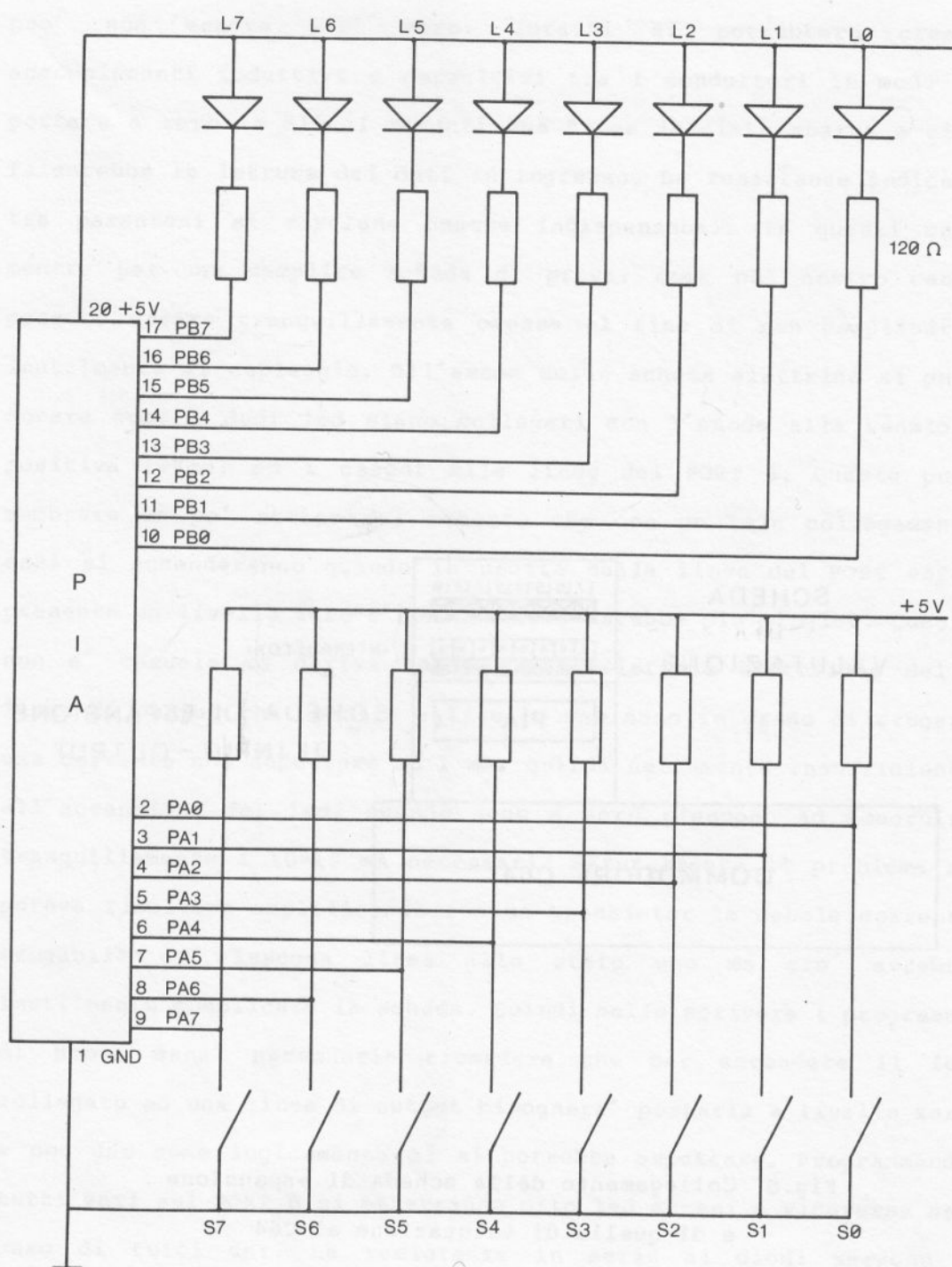



Fig.7 Schema elettrico della scheda di valutazione delle funzioni di I/O del PIA connesso al C64

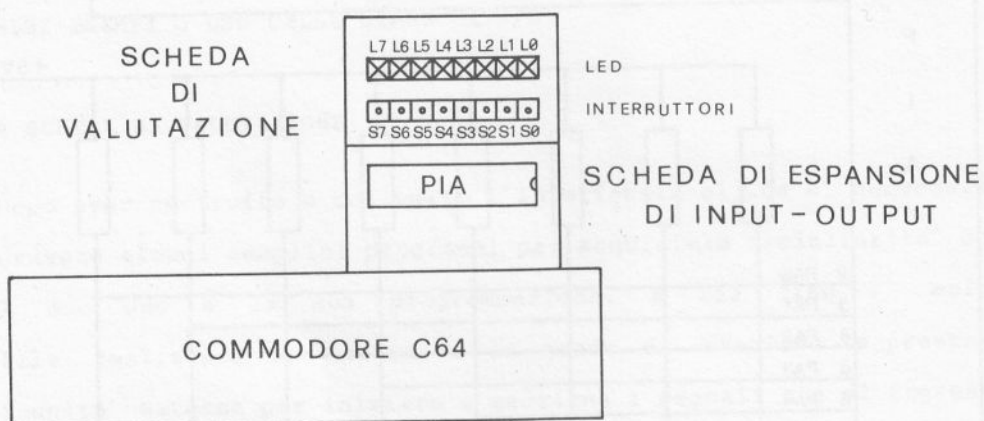



Fig.8 Collegamento della scheda di espansione e di quella di valutazione al C64

puo′ non essere piu′ vero. Infatti si potrebbero creare accoppiamenti induttivi o capacitivi tra i conduttori in modo da portare a zero in alcuni momenti una linea lasciata aperta e cio′ falserebbe la lettura dei dati in ingresso. Le resistenze indicate tra parentesi si rivelano dunque indispensabili in questi casi mentre per una semplice scheda di prova, come nel nostro caso, possono essere tranquillamente omesse al fine di non complicarne inutilmente il cablaggio. Dll′esame dello schema elettrico si puo′ notare come i dodi led siano collegati con l′anodo alla tensione positiva (+Vcc) ed i catodi alle linee del PORT B. Questo puo′ sembrare un po′ strano dal momento che con un tale collegamento essi si accenderanno quando in uscita dalla linea del PORT sara′ presenta un livello zero e non uno come sarebbe piu′ logico. Questo non e′ casuale ma deriva dalle caratteristiche elettriche delle linee di output che, mentre a livello uno sono in grado di erogare una corrente non superiore ad 1 mA, quindi nettamente insufficiente all′accensione del led, quando sono a zero riescono ad assorbire tranquillamente i 10-15 mA necessari. Naturalmente il problema si poteva risolvere amplificando con un transistor la debole corrente erogabile da ciascuna linea allo stato uno ma cio′ avrebbe inutilmente complicato la scheda. Quindi nello scrivere i programmi di prova sara′ necessario ricordare che per accendere il led collegato ad una linea di output bisognera′ portarla a livello zero e non uno come logicamente ci si potrebbe aspettare. Programmando tutti zeri sul PORT B si otterranno otto led accesi e viceversa nel caso di tutti uni. Le resistenze in serie ai diodi servono a limitarne la corrente di accensione a livelli accettabili per le linee del PIA anche se in teoria potrebbero essere omesse o sostituite con una sola di valore minore sulla linea comune degli anodi.

```
1 rem visualizzazione del numero
2 rem impostato sulle linee del port a
3 rem
10 gosub1000:rem inizializzazione porte
20 printpeek(x);:rem valore letto
30 goto20:rem da capo
90 rem
95 rem output di un numero sulle
98 rem linee del port b
99 rem
100 gosub1000:rem inizializzazione
105 pokex+2,255:rem pulizia linee di out
110 input"valore in output (0-255) ";n
115 ifn<0orn>255then:print"ho detto tra 0 e 255 !":goto110
120 pokex+2,255-n:rem output valore
130 goto110:rem da capo
140 rem
1000 rem inizializzazione porte******
1010 x=13*16↑3+15*16↑2+8*16+4
1019 rem**scopre ddra e ddrb
1020 pokex+1,0:pokex+3,0
1029 rem**00000000 in ddra e 11111111 in ddrb
1030 pokex,0:pokex+2,255
1038 rem**scopre ora e orb*****
1040 pokex+1,4:pokex+3,4
1049 rem**inizializzazione finita****
1050 return
```

PRG5.Input e output con le porte del
PIA 6821

```
1 rem programma per la rotazione di
2 rem di un led acceso su 64 led
3 rem ***albero di natale*******
4 rem
10 gosub1000:rem inizializza i port
11 pokex,0:pokex+2,255
13 forg=0to7:rem per gli otto gruppi
14 pokex,2↑g
15 rem*****da destra a sinistra******
20 forn=0to7:rem per gli otto bit
30 pokex+2,255-2↑n:rem  accende un led alla volta
35 fort=0to20:nextt:rem loop di attesa
40 nextn:nextg
41 pokex,0:pokex+2,255
42 forg=7to0step-1:rem per gli otto gruppi
44 pokex,2↑g
45 rem*****da sinistra a destra******
50 forn=7to0step-1
60 pokex+2,255-2↑n
70 fort=0to20:nextt
80 nextn:nextg
99 goto11:rem da capo
1000 rem inizializzazione porte******
1010 x=13*16↑3+15*16↑2+8*16+4
1019 rem**scopre ddra e ddrb
1020 pokex+1,0:pokex+3,0
1029 rem**11111111 in ddra e 11111111 in ddrb
1030 pokex,255:pokex+2,255
1038 rem**scopre ora e orb*****
1040 pokex+1,4:pokex+3,4
1049 rem**inizializzazione finita****
1050 return
```

PRG6. Esempio di uso dell'interfaccia per illuminare l'albero natalizio con 64 LED.

```
1 rem programma per la rotazione di
2 rem di un led spento sul port b
3 rem *******ping-pong**********
10 gosub1000:rem inizializza i port
15 rem*****da destra a sinistra******
20 forn=0to7:rem per gli otto bit
30 pokex+2,2↑n:spegne un led alla volta
35 fort=0to20:nextt:rem loop di attesa
40 next
45 rem*****da sinistra a destra******
50 forn=7to0step-1
60 pokex+2,2↑n
70 fort=0to20:nextt
80 next
99 goto20:rem da capo
1000 rem inizializzazione porte******
1010 x=13*16↑3+15*16↑2+8*16+4
1019 rem**scopre ddra e ddrb
1020 pokex+1,0:pokex+3,0
1029 rem**00000000 in ddra e 1111111 in ddrb
1030 pokex,0:pokex+2,255
1038 rem**scopre ora e orb*****
1040 pokex+1,4:pokex+3,4
1049 rem**inizializzazione finita****
1050 return
```

PRG7.Esempio di gestione delle linee dei port del PIA 6521

```
 1 rem movimento di due led accesi
 2 rem a velocita' controllabile con
 3 rem i tasti + e -
 4 rem ********auguri***********
 5 rem
 9 gosub2000:rem scrive "a u g u r i !"
 10 gosub 1000:inizilaizza i port
 20 forn=0to7:rem per ogni bit
 30 pokex+2,(255-2↑n-2↑(7-n))
 33 gosub100:rem test per cambio velocita'
 35 fort=0tov:nextt:rem loop di attesa
 40 next
 45 rem *****da capo*******
 50 forn=7to0step-1
 60 pokex+2,(255-2↑n-2↑(7-n))
 65 gosub100
 70 fort=0tov:nextt
 80 next
 89 goto20
 90 rem
 95 rem cambio velocita'
 99 goto20
100 getc$:ifc$=""then150
110 ifc$="+"then:v=v+10
120 ifc$="-"then:v=v-10
150 return
1000 rem inizializzazione porte******
1010 x=13*16↑3+15*16↑2+8*16+4
1019 rem**scopre ddra e ddrb
1020 pokex+1,0:pokex+3,0
1029 rem**00000000 in ddra e 1111111 in ddrb
1030 pokex,0:pokex+2,255
1038 rem**scopre ora e orb*****
1040 pokex+1,4:pokex+3,4
1049 rem**inizializzazione finita****
1050 return
2000 rem** a u g u r i !*******
2005 print"S"
2010 print:print:print
2011 print
2020 print"   *   *   *  *** *   * ***  *  *
2021 print"  * *  *   * *    *   * *  * *  *
2022 print" *   * *   * *    *   * *  * *  *
2023 print" *   * *   * *    *   * * *  *  *
2024 print" ***** *   * * ** *   * **   *  *
2025 print" *   * *   * *  * *   * * *  *  *
2026 print" *   * *   * *  * *   * *  * *
2027 print" *   * ***** **** ***** *  * *  *
2028 print
2030 return
```

PRG8. Esempio di uso dell'interfaccia
di I/O per giochi luminosi su 8
LED

PROGRAMMI DI PROVA

Dai programmi indicati e' agevole verificare la semplicita' di uso delle linee di I/O del 6521 in quanto, dopo averne programmata la funzione, input o output, esse vengono trattate come bit di una qualunque locazione di memoria. PRG5 serve semplicemente per leggere il numero impostato sugli interruttori di ingresso mentre con RUN 100 e' possibile visualizzare sui led lo stato delle linee di output scrivendo nel PORT B qualunque numero compreso tra 0 (tutte accese) e 255 (tutte spente). Se invece di mandare in uscita il numero voluto N si invia (255-N) si ottiene il risultato opposto e cioe' risulteranno accesi solo i led corrispondenti a bit ad uno e spenti i rimanenti. I programmi PRG7 e PRG8 forniscono semplici esempi di manipolazione delle linee di output al fine di ottenere effetti luminosi sui led e rappresentano esempi di gestione delle linee di output. PRG8 puo' essere usato per creare una simpatica illuminazione dell'albero di Natale casalingo disponendo oportunamente gli otto led non piu' sulla scheda ma sui rami.

GESTIONE MULTIPLEXED DI OUTPUT

L'esempio di PRG8 puo' servire ad illuinare solo piccoli alberi natalizi disponendo solo di otto led mentre volendo ottenere un effetto migliore ne sarebbero necessari molti di piu'. Sfruttando come output anche le linee del PORT A, CA2 e CB2 si puo' arrivare pilotare sino a 18 led ma non oltre. Certamente collegando al C64 altri chip di I/O nel modo visto e' possibile arrivare al numero di output voluti ma complicando notevolmente il circuito elettrico. Esiste pero' una tecnica che permette di gestire ben 64 linee con un solo chip di I/O detta "MULTIPLEXED" che potra' rivelarsi molto utile in una grande varieta' di applicazioni. Con riferimento allo

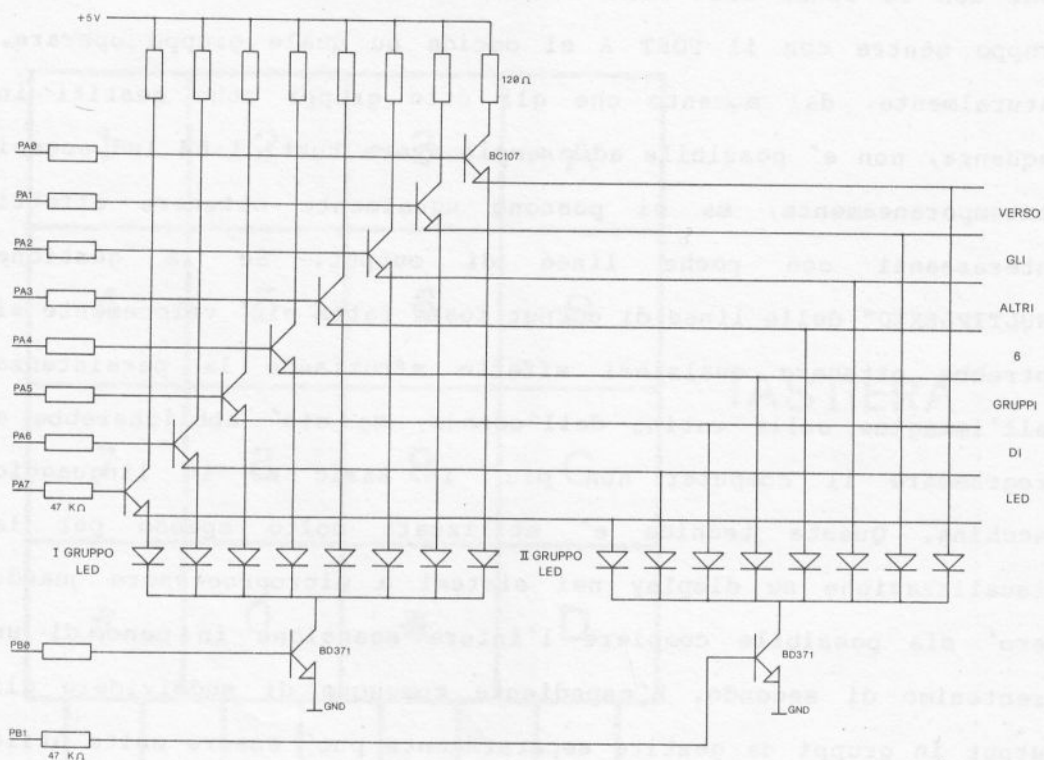



Fig.9 Gestione "MULTIPLEXED" di output sino ad un massimo di 8 gruppi di 8 led ciascuno (totale 64) con 16 linee

schema elettrico si puo' notare che i led sono ora suddivisi in otto gruppi ciascuno dei quali ne contiene otto, in tutto dunque 64 led. Con le linee del PORT B si abilitano i led di un singolo gruppo mentre con il PORT A si decide su quale gruppo operare. Naturalmente, dal momento che gli otto gruppi sono gestiti in sequenza, non e' possibile ad esempio avere tutti i 64 led accesi contemporaneamente, ma si possono ugualmente ottenere effetti interessanti con poche linee di output. Se la gestione "MULTIPLEXED" delle linee di output fosse fatta piu' velocemente si potrebbe ottenere qualsiasi effetto sfruttando la persistenza dell'immagine sulla retina dell'occhio, ma cio' obbligherebbe a programmare il computer non piu' in Basic ma in linguaggio macchina. Questa tecnica e' utilizzata molto spesso per la visualizzazione su display nei sistemi a microprocessore quando pero' sia possibile compiere l'intera scansione in meno di un trentesimo di secondo. L'espediente comunque di suddividere gli output in gruppi da gestire separatamente puo' essere molto utile nei casi in cui il numero di segnali sia molto numeroso e la velocita' di elaborazione sufficientemente elevata.

USO DELLE LINEE IN INPUT

Per trovare una interessante applicazione della scheda di espansione di input/output usata come ingresso conviene riprendere il discorso iniziato precedentemente riguardo ad un sistema anti-furto per abitazioni. Si era giunti allora a concludere che il semplice controllo della ciusura delle vie di accesso alla casa non poteva essere sufficiente per raggiungere il nostro obiettivo. Diventava, ad esempio, molto arduo per il padrone riuscire ad escludere l'allarme prima di rientrare onde non caderne vittima lui stesso. Si era pensato di ritardare l'attivazione della suoneria di

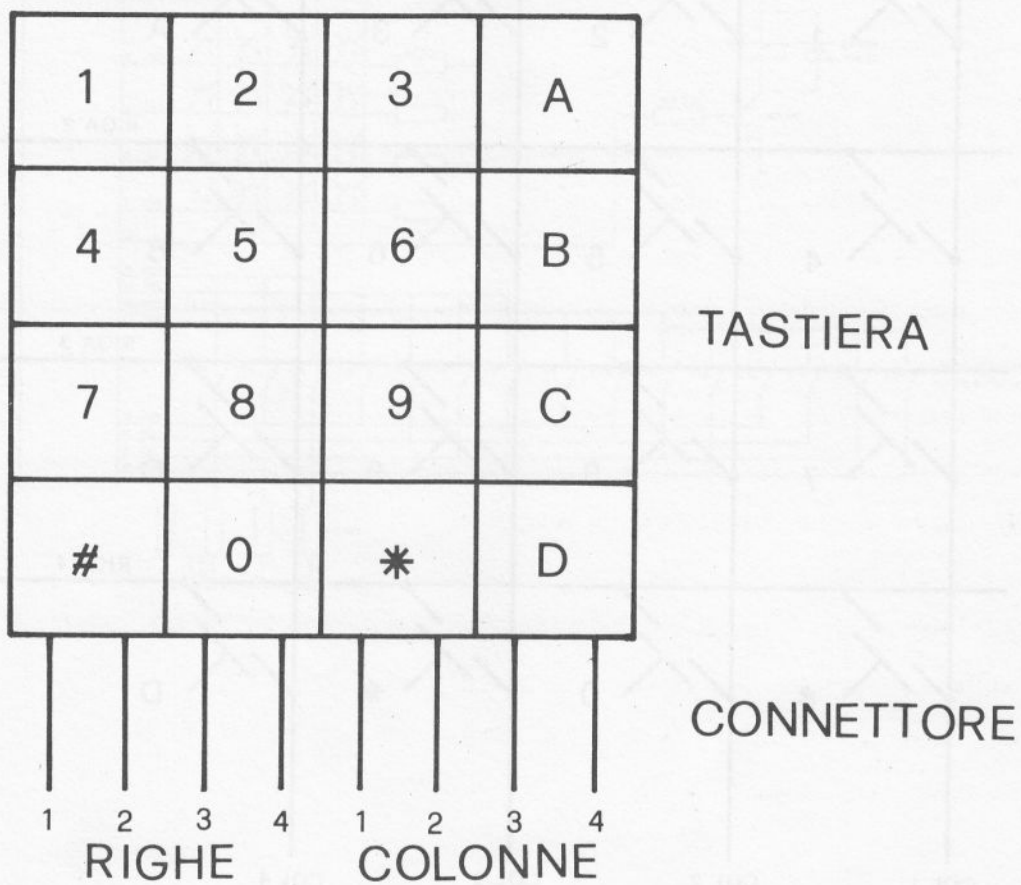



Fig.10 Tastiera di tipo telefonico a 16 tasti

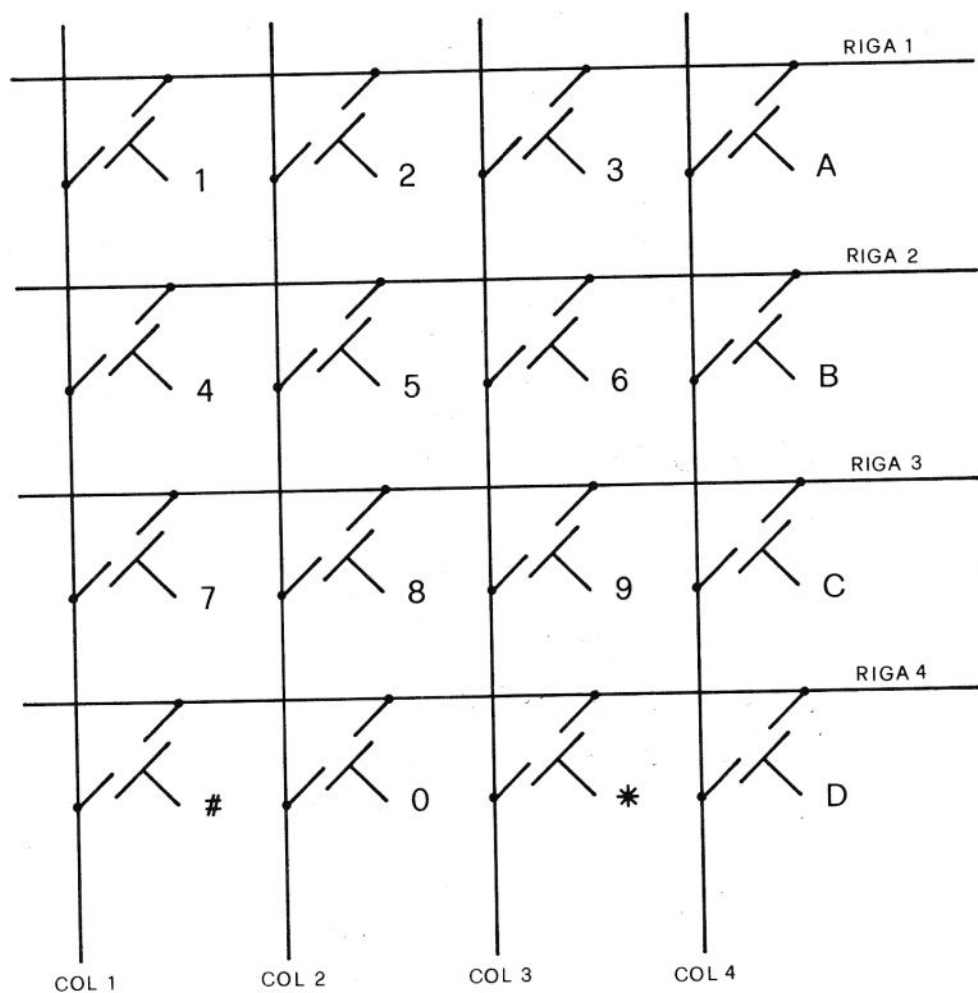



Fig.11 Schema elettrico della tastiera telefonica a 16 tasti. La struttura e' a matrice 4 X 4 (4 righe per 4 colonne). Ogni incrocio individua univocamente un tasto

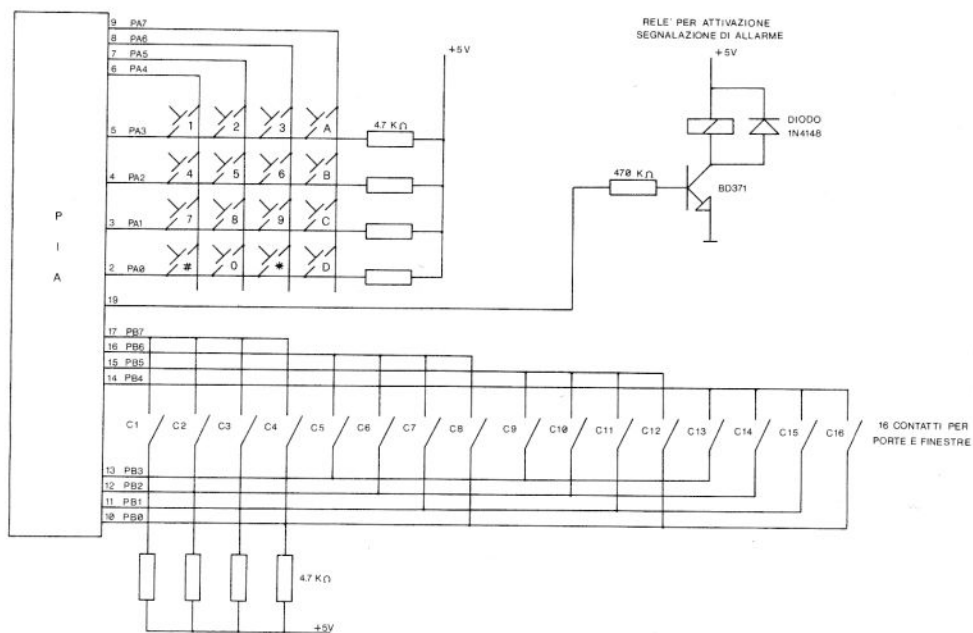



Fig.12 Circuito elettrico completo relativo al sistema di allarme a combinazione impostabile con tastiera alfanumerica

```
1 rem *********uso della tastiera****
2 rem *********esterna per il *******
3 rem *********riconoscimento di ****
4 rem *********una chiave  **********
5 rem *********chiave da indovinare**
8 co$="a2233" :rem a scelta
9 rem
10 gosub1000 :rem inizializzazione tastiera
15 pokex,240
18 gosub2000 :rem tabella dei tasti
20 gosub3000 :rem lettura tasto premuto
30 ifcc$<>""then:printcc$;:rem carattere letto
40 ci$=ci$+cc$ :rem accumula nella chiave
50 iflen(ci$)<>len(co$)then20 :rem chiave completa ?
60 ifci$=co$then:print"ok ",co$,ci$ :rem ok indovinato
70 ifci$<>co$then:print"ko",co$,ci$ :rem peccato
80 ci$="":goto20 :rem da capo
99 end
1000 rem inizializzazione porte******
1010 x=13*16↑3+15*16↑2+8*16+4
1019 rem**scopre ddra e ddrb
1020 pokex+1,0:pokex+3,0
1028 rem**scopre ora e orb*****
1029 rem**00000000 in ddra e 1111111 in ddrb
1030 pokex,240:pokex+2,255
1040 pokex+1,4:pokex+3,4
1049 rem**inizializzazione finita****
1050 return
1998 rem
1999 rem
2000 rem inizializzazione tastiera
2010 dimta$(15):pokex,240
2020 fork=0to15
2030 readta$(k)
2040 next
2050 return
2060 data 1,2,3,a
2070 data 4,5,6,b
2080 data 7,8,9,c
2090 data #,0,*,d
3000 rem lettura tastiera
3010 cc$=""
3020 l=128
3030 fork=0to16step4
3040 pokex,255-l
3050 t=peek(x)and15:ift=15then:c=c+1:goto3068
3051 rem   decodifica tasto
3052 ift=7thenj=3
3054 ift=11thenj=2
3056 ift=13thenj=1
3057 ift=14thenj=0
3059 ifst$="§"then3068
3065 cc$=ta$(k+j)
3068 l=l/2
3070 next
3075 ifc=5then:st$="":c=0:goto3099
3080 st$="§":c=0
3099 return
```

PRG9 . Programma per usare una
       tastiera esterna come chiave
       elettronica in un sistema
       antifurto domestico

```
1 rem *********sistema di allarme****
2 rem *********a chiave alfanumerica*
3 rem *********escludibile da *******
4 rem *********tastiera**************
9 rem
10 gosub1000 :rem inizializzazione tastiera
12 rem              e ingressi di allarme
13 gosub800: rem introduzione codice
15 pokex,240:pokex+2,240
18 gosub2000 :rem tabella dei tasti
20 gosub3000 :rem lettura tasto premuto
40 ci$=ci$+cc$ :rem accumula nella chiave
50 iflen(ci$)<>len(co$)then100 :rem chiave completa ?
60 ifci$=co$then:al=1 :rem ok indovinato
70 ifci$<>co$then:al=0 :rem peccato
80 ci$="" :rem continua
100 gosub4000: rem controllo ingressi
110 ifal=1then200:rem allarme disattivato
115 ifaa=0then200:rem appena attivato
120 ifaa=1then200:rem accetta una apertura
125 rem                per permettere l'uscita
130 pokex+1,4+32+16: rem allarme in funzione
135 print"sqqqqqqqqqqÜÜÜÜÜrallarme in funzione !!!!!R"
140 goto20
200 pokex+1,4+32+16+8: rem allarme spento
205 print"sqqqqqqqqqqÜÜÜÜÜ
210 goto20
798 rem
799 rem
800 rem definizione chiave di accesso
810 print"S":input"qqqqqqÜÜÜÜchiave ";co$
820 print"qqqqqok !!"
830 fori=0to700:next
840 print"S":print"          qqqqqqrallarme attivoR"
845 al=0:rem allarme attivo
850 return
998 rem
999 rem
1000 rem inizializzazione porte******
1010 x=13*16↑3+15*16↑2+8*16+4
1019 rem**scopre ddra e ddrb
1020 pokex+1,0:pokex+3,0
1028 rem**scopre ora e orb*****
1029 rem**00000000 in ddra e 11111111 in ddrb
1030 pokex,240 :rem port a per tastiera
1035 pokex+2,240 :rem port b per allarmi
1040 pokex+1,4+32+16+8:pokex+3,4
1049 rem**inizializzazione finita****
1050 return
1998 rem
1999 rem
2000 rem inizializzazione tastiera
2010 dimta$(15):pokex,240
2020 fork=0to15
2030 readta$(k)
2040 next
2050 return
2060 data 1,2,3,a
2070 data 4,5,6,b
```

```
2080 data 7,8,9,c
2090 data #,0,*,d
3000 rem lettura tastiera
3010 cc$=""
3020 l=128
3030 fork=0to16step4
3040 pokex,255-l
3050 t=peek(x)and15:ift=15then:c=c+1:goto3068
3051 rem   decodifica tasto
3052 ift=7thenj=3
3054 ift=11thenj=2
3056 ift=13thenj=1
3057 ift=14thenj=0
3059 ifst$="§"then3068
3065 cc$=ta$(k+j)
3068 l=l/2
3070 next
3075 ifc=5then:st$="":c=0:goto3099
3080 st$="§":c=0
3099 return
3998 rem
3999 rem
4000 rem lettura contatti di allarme
4010 l=128
4020 fork=0to3
4030 pokex+2,255-l
4040 t=peek(x+2)and15
4050 ift<>0then:aa=aa+1:k=4
4060 l=l/2:next
4070 return
```

PRG10. Programma per usare il C64
in un completo sistema di
allarme per abitazione con
esclusione a chiave alfanumerica

allarme onde permetterne l'esclusione ma anche cio' non era stato giudicato molto brillante. A questo punto pero' e' possibile completare il sistema con una chiave elettronica cioe' con una tastiera sulla quale poter digitare un codice noto soltanto alle persone autorizzate in modo da escludere o attivare a piacimento il sistema di allarme. Praticamente sulla porta di ingresso o immediatamente vicino trovera' posto una tastiera a 16 contatti, di tipo telefonico ad esempio, sulla quale dovra' essere impostata una precisa sequenza di numeri e lettere, modificabile a piacere sia nel contenuto che nella lunghezza, mediante la quale il computer sara' in grado di decidere se escludere la protezione o meno degli accessi alla casa. Tali tastiere sono facilmente reperibili e sono realizzate anche a membrana, adatte dunque per essere installate in luoghi esterni o comunque esposti ad umidita' ed intemperie. La connessione con il computer avviene tramite solo otto linee, come e' indicato nello schema, dal momento che la lettura e' eseguita non per tasti singoli (16) ma per righe (4) e colonne (4) occupando in tal modo solo un port del PIA di espansione. Il rimanente port e' usato per controllare l'apertura delle vie di accesso alla casa ed una ulteriore linea per l'attivazione dell'allarme acustico o luminoso. Il tipo di gestione della tastiera detto anche "a matrice" merita qualche considerazione in piu' dal momento che viene utilizzato anche con tipi di tastiere dotate di un numero maggiore di contatti, ad esempio quella che equipaggia il computer stesso. Essa infatti permette, con un limitato numero di linee di ingresso, la scansione di un grande numero di tasti. Nel caso in esame si hanno quattro linee programmate come ingressi, ciascuna delle quali collegata ai tasti di una stessa colonna, ad esempio 1,4,7,# o A,B,C,D e quattro linee in output collegate alle righe della tastiera. Ciascun tasto

si trova dunque all'incrocio di una delle linee di input con una di quelle di output. Portando a zero in sequenza una sola alla volta delle uscite (righe) e controllando ogni volta se qualcuna di quelle in input (colonne) si trova allo stesso livello logico zero si riesce ad individuare quale tasto e' premuto. Con una opportuna routine viene quindi associato ad ogni incrocio di linee il carattere corrispondente, lettera o numero a secondo della tastiera adottata. Quanto visto viene eseguito qualche centinaio di volte al secondo nel programma di esempio indicato. Sempre nel programma e' previsto che un tasto, una volta trovato premuto venga accettato solo una volta anche se rimane premuto a lungo. Cioe' dopo ogni pressione di un tasto la tastiera deve rimanere libera almeno per il tempo necessario ad una intera scansione prima di accettare il successivo. Questa tecnica si rivela ancora piu' efficacie con tastiere grandi permettendo, ad esempio, di gestire sino a 64 tasti con solo 16 linee delle quali otto di ingresso e otto di uscita. Sempre a proposito del sistema anti-furto si puo' adottare la stessa tecnica per rilevare l'apertura di una delle vie di accesso all'abitazione. Infatti, dal momento che sulla scheda di espansione rimangono solo otto linee libere, si potrebbero avere problemi dovendo sorvegliare un numero di punti superiore. Dal momento che l'apertura di una porta o di una finestra corrisponde all' apertura di un contatto di un rele' magnetico e' possibile collegare quest'ultimi a matrice riuscendo a gestirne ben 16 con le linee ancora disponibili. Le linee di connessione tra il computer e la tastiera cosi' come i contatti dei rele' potranno avere una discreta lunghezza, dell' ordine di qualche decina di metri, senza creare problemi. Dovendo coprire distanze maggiori sara' necessario accoppiare alla scheda di espansione un'altre a rele' onde evitare disturbi che potrebbero nuocere sia al funzionamento che

all'integrita' elettrica dell'intero sistema di elaborazione.

AUTO-START
PER COMMODORE 64

INTRODUZIONE

Il basso prezzo e la grande facilita' d'uso hanno reso il C64 uno dei piu' diffusi personal computer nel mondo. La grande varieta' di software disponibile permette di utilizzarlo in ambiti sempre piu' vasti al di fuori di quello didattico o casalingo per il quale sembrava essere stato pensato. Frequentemente si trovano tali sistemi utilizzati in applicazioni specifiche, con programmi sviluppati appositamente non solo nel campo dell'elaborazione dei dati ma anche nella loro acquisizione tramite opportune interfacce. Gli utenti finali di tali sistemi dedicati non sono normalmente programmatori e quindi mal si adattano ad imparare la procedura di avviamento del computer, il caricamento del programma e tutto il resto. L'ideale sarebbe che il C64 all'accensione caricasse automaticamente il programma ed iniziasse l'esecuzione immediatamente. Nel caso poi di sistemi dedicati alla rilevazione dei valori di grandezze fisiche (temperatura, pressione, ecc), operazioni che possono durare anche molte ore, una interruzione anche brevissima della tensione di rete e' sufficiente per resettare il computer ed a interrompere l'acquisizione finche' qualcuno non se ne accorge e non ricarica il programma. Come si puo' vedere un sistema di partenza automatica (auto-start) puo' veramente trasformare un C64 in un sistema dedicato "chiavi in mano", nel senso che basta inserire il disco con il programma, accendere l'unita' centrale ed iniziare a lavorare senza che l'operatore debba ricordare o digitare alcunche'.

PRINCIPIO DI FUNZIONAMENTO

Ognuno avra' avuto modo di constatare che inserendo nell'opportuno connettore la cartuccia di un gioco e accendendo il C64 esso inizia immediatamente ad eseguirlo senza necessita' di alcuna operazione manuale. Questo significa che in qualche modo il computer si accorge della presenza del software aggiunto ed inizia ad eseguirlo senza attivare il Basic ed il sistema operativo consueto. Infatti la routine di Reset ($FCE2) per prima cosa controlla che nei primi indirizzi ($8003-$8008) dell'area destinata al software esterno ($8000-$9FFF) sia presente una sequenza precisa di caratteri ed in caso affermativo interrompe la normale gestione cedendo il controllo alla routine il cui indirizzo si trova agli indirizzi $8000-$8001. Tale sequenza di identificazione e' costituita da 5 caratteri e precisamente da C B M 8 0 con le lettere codificate in codice ASCII ma con il bit di peso maggiore ad uno. E' estrememente improbabile che all'accensione una tale sequenza si venga a trovare casualmente nella ram interna quindi il C64 proseguira' nella sua normale inizializzazione. Viceversa qualunque cartuccia che voglia essere presa in considerazione dal sistema dovra' averla nelle locazioni richieste ed avere nelle sue prime due locazioni l'indirizzo di inizio del programma stesso. Le locazioni terza e quarta dovranno contenere l'indirizzo della routine da eseguire in caso di ripartenza a caldo, in pratica cio'che si verifica premendo contemporaneamente RUN e RESTORE. Naturalmente la ROM esterna si sostituisce alla RAM interma tra gli indirizzi $8000 e $9FFF occupando in tutto 8 Kbytes. Il programma di auto-start, di cui si fornisce il listato in linguaggio Assembler, e' molto semplice ma pur essendo cortissimo sottrae purtroppo gli ultimi 8 Kbytes di RAM alla successiva gestione Basic per quanto detto sopra. In pratica esso simula la sequenza di reset

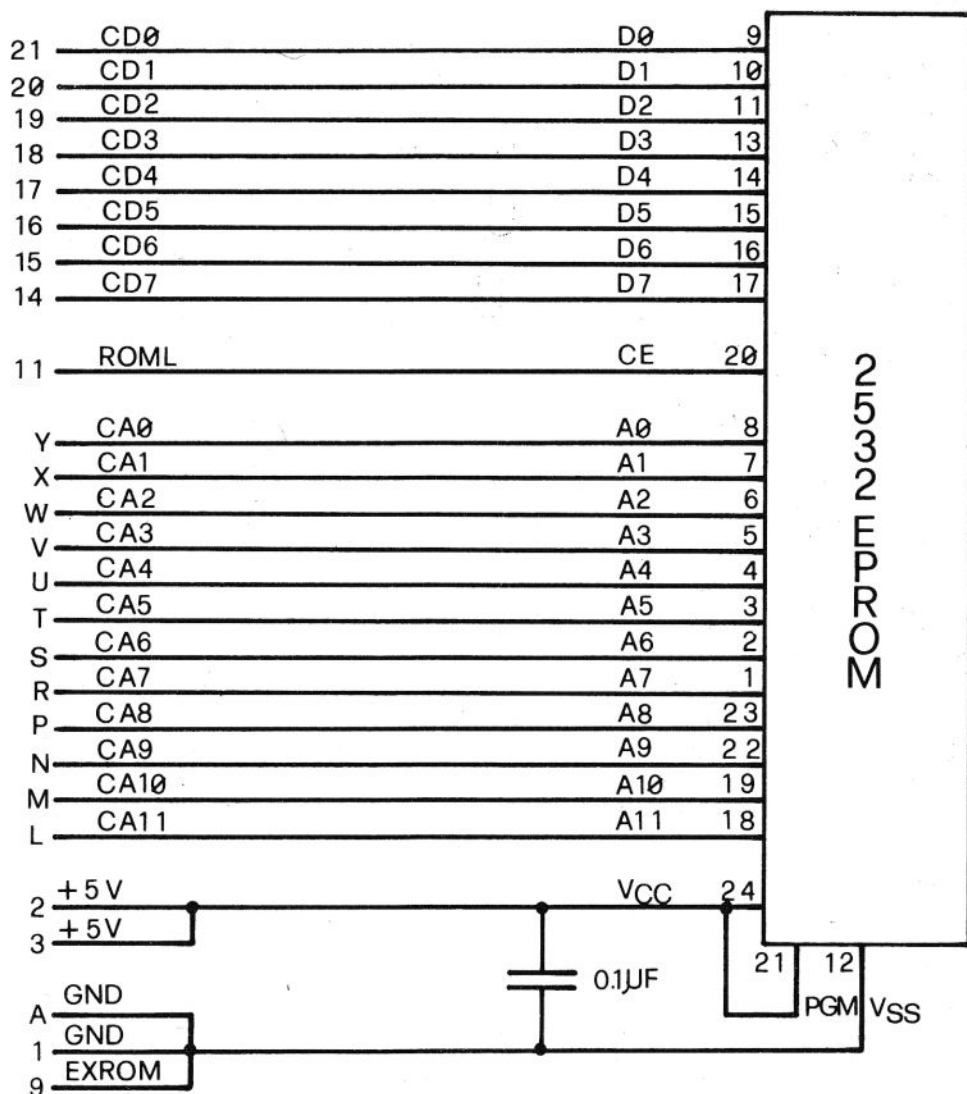



Fig.13 Connessione della Eprom di AUTO-START

al connettore di espansione del C64

tipica del sistema operativo del C64 mettendo pero' nel buffer della tastiera quello che l'operatore avrebbe dovuto digitare appena dopo l'accensione e cioe' LOAD"*",8 ed il successivo RUN. Il carattere asterisco, come noto, provoca il caricamento del primo programma presente sul disco ed il RUN lo manda in esecuzione. Evidentemente chi dovesse eseguire una fase di inizializzazione piu' lunga, magari con diversi POKE, potra' aggiungerla al posto giusto e tutto verra' eseguito correttamente. La parte rimanente della ROM, circa 4 Kbyte potra' contenere routine di utilita' in linguaggio macchina per non sprecare tanta memoria inutilmente.

LA SCHEDA

Come si puo' vedere dallo schema la scheda da aggiungere al C64 e' molto semplice e potra' essere realizzata anche a Wire-Wrap su una piccola scheda forata. L'unico vero problema sara' quello di reperire una scheda sperimentale con il connettore adatto a quello dotato di 22+22 contatti che si trova sul lato posteriore del C64. Nei casi disperati si puo' sempre riutilizzare la scheda contenuta nella cartuccia di un gioco che non ci diverte piu' dissaldando con cura la ROM e sostituendola con uno zoccolo da 24 pin. Il programma dovra' naturalmente essere memorizzato su una memoria tipo TMS2532 o TMS2516 con un circuito programmatore di Eprom.

```
AUTO0......PAGE 0001

LINE# LOC   CODE        LINE

00001  0000             ;          ***********************
00002  0000             ;          *    ROUTINES DI      *
00003  0000             ;          *  AUTO-START PER     *
00004  0000             ;          *  COMMODORE  C64     *
00005  0000             ;          ***********************
00006  0000             ;
00007  0000             ;
00008  0000             ORIG=$8000                ;INDIRIZZO CARTUCCIA
00009  0000             ;
00010  0000             BUFTA=$277                ;INDIRIZZO BUFFER TASTIERA
00011  0000             COUNT=$C6                 ;COUNTER CARATTERI NEL BUFFER
00012  0000             ;
00013  0000                    *=ORIG             ;INIZIO PROGRAMMA
00014  8000  09 80             .WOR START         ;VETTORE DI INIZIO A FREDDO
00015  8002  5E FE             .WOR $FE5E         ;    "          "    CALDO
00016  8004  C3                .BYT $C3,$C2,$CD,$38,$30 ;IDENTIFICATORE CARTUCCIA
00016  8005  C2
00016  8006  CD
00016  8007  38
00016  8008  30
00017  8009                                       ;C+$80,B+$80,M+$80,8,0
00018  8009             ;
00019  8009  8E 16 D0   START  STX $D016          ;ATTIVA IL CONTROLLER VIDEO
00020  800C  20 A3 FD          JSR $FDA3          ;INIZIALIZZA LE UNITA' DI I/O
00021  800F  20 50 FD          JSR $FD50          ;    "       I PUNTATORI DI MEMORIA
00022  8012  20 15 FD          JSR $FD15          ;RIPRISTINA I VETTORI DI I/O
00023  8015  20 5B FF          JSR $FF5B          ;INIZIALIZZA SCHERMO E TASTIERA
00024  8018  20 53 E4          JSR $E453          ;INIZIALIZZA I VETTORI DEL
00025  801B                                       ;DEL SISTEMA OPERATIVO
00026  801B  20 BF E3          JSR $E3BF          ;ATTIVA L'INTERPRETER BASIC
00027  801E  A9 06             LDA #$06           ;DEFINISCE IL COLORE DEI CARATTERI
00028  8020  8D 86 02          STA $286           ;NELLA LOCAZIONE COMUNE
00029  8023  A2 00             LDX #0
00030  8025  BD 38 80   LOPCH  LDA TABA,X         ;PRENDE IL CARATTERE
00031  8028  F0 08             BEQ VAICH          ;SE ZERO ALLORA FINITO
00032  802A  9D 77 02          STA BUFTA,X        ;LO METTE NEL BUFFER TASTIERA
00033  802D  E8                INX                ;SUCCESSIVO CARATTERE
00034  802E  86 C6             STX COUNT          ;INCREMENTE IL CONTATORE DEI CARATTERI
00035  8030  D0 F3             BNE LOPCH          ;CONTINUA
00036  8032  A2 FB      VAICH  LDX #$FB           ;RIPRISTINA LO STACK-POINTER
00037  8034  9A                TXS                ;AL VALORE DI PARTENZA
00038  8035  4C 7B A4          JMP $A47B          ;ED INIZIA AD ESEGUIRE
00039  8038             ;
00040  8038  4C 4F      TABA   .BYT 'LOAD"*",8'
00041  8041  0D                .BYT $0D           ;CARRIAGE RETURN
00042  8042  52 55 4E          .BYT 'RUN'
00043  8045  0D                .BYT $0D
00044  8046             ;AGGIUNGERE QUI ALTRE EVENTUALI ISTRUZIONI
00045  8046  00                .BYT 0             ;FINE COMANDI
00046  8047                    .END

ERRORS = 00000
```

ESPANSIONE SERIALE DI I/O
PER COMMODORE C64

INTRODUZIONE

Il computer, ogni giorno di piu', trova applicazione nel controllo automatico di processi fisici dai piu' semplici ai piu' complessi permettendone gestioni anche molto sofisticate o come si suol dire "intelligenti" con vantaggi che chiunque e' in grado di apprezzare. Gli ingressi possono ad esempio essere costituiti da segnali provenienti da termostati o da misuratori di livello oppure segnali di fine-corsa meccanici o magnetici. In pratica ciscuna linea segnala se una certa grandezza ha superato o meno un valore prefissato o se un evento si e' verificato. Le uscite di un sistema potrebbero essere segnali di avviamento di motori oppure di riscaldatori o pompe oppure indicatori di allarme. Come si puo' vedere e' possibile controllare un grande numero di sistemi fisici per mezzo di un computer, definendo opportunamente il numero e la natura dei segnali di ingresso e di uscita. Si potrebbe anche osservare che la presenza del computer potrebbe non rivelarsi indispensabile dal momento che ad esempio un termostato puo' attivare o disattivare direttamente un riscaldatore cosi' come un pressostato per una pompa oppure un contatto fine-corsa puo' comandare direttamente l'arresto di un motore. Questo e' vero quando ogni variabile del sistema si puo' considerare indipendente dalle altre e dal tempo mentre diversamente il discorso si puo' presentarsi notevolmente piu' complesso. Infatti se, in una certa applicazione, l'attivazione di un compressore dipende sia dalla pressione che dalla temperatura e deve durare tempi diversi a seconda della fase di lavorazione in atto, la presenza di un

controllore intelligente comincia a rivelarsi indispensabile. Si pensi come ulteriore esempio al condizionamento di diversi locali in funzione dell′ ora della giornata e del giorno della settimana, tenendo conto anche delle festivita′ infrasettimanali, per rendersi conto che ben difficilmente il problema puo′ essere risolto senza l′ausilio di un computer.

Cio′ premesso il problema potrebbe anche apparire di facile soluzione dal momento che quasi tutti i computer, anche i piu′ semplici, sono dotati di porte di input/output in grado sia di leggere lo stato logico (zero/uno) di linee esterne sia di inviare comandi a unita′ esterne sotto il controllo di un programma. Chi si sia cimentato nella realizzazione di un sistema di controllo utilizzando un personal tipo il Commodore C64 avra′ dovuto probabilmente scontrarsi con tutta una serie di problemi elettrici tali da rendere spesso inattuabile in pratica un progetto teoricamente ineccepibile. Il primo e piu′ banale di essi e′ rappresentato dal numero delle linee di I/O disponibili nel computer che difficilmente superano la decina mentre anche per applicazioni banali le necessita′ sono spesso molto superiori. Il secondo, molto piu′ insidioso, e costituito dal fatto che difficilmente il sistema di controllo puo′ essere posto a breve distanza (poche decine di centimetri) dal processo da controllare e quindi i fili di connessione degli ingressi sono costretti a lunghi e pericolosi percorsi in ambienti elettricamente poco salubri, ricchi cioe′ di disturbi, che non solo possono falsare i valori letti ma a volte riescono ad essere fatali per i chip di I/O, non certo robustissimi, dei quali e′ dotato l′elaboratore. La necessita′ di prolungare la massa elettrica del computer per diverse decine di metri magari vicino a grossi motori elettrici o teleruttori di potenza puo′, un giorno o l′ altro, provocarne una

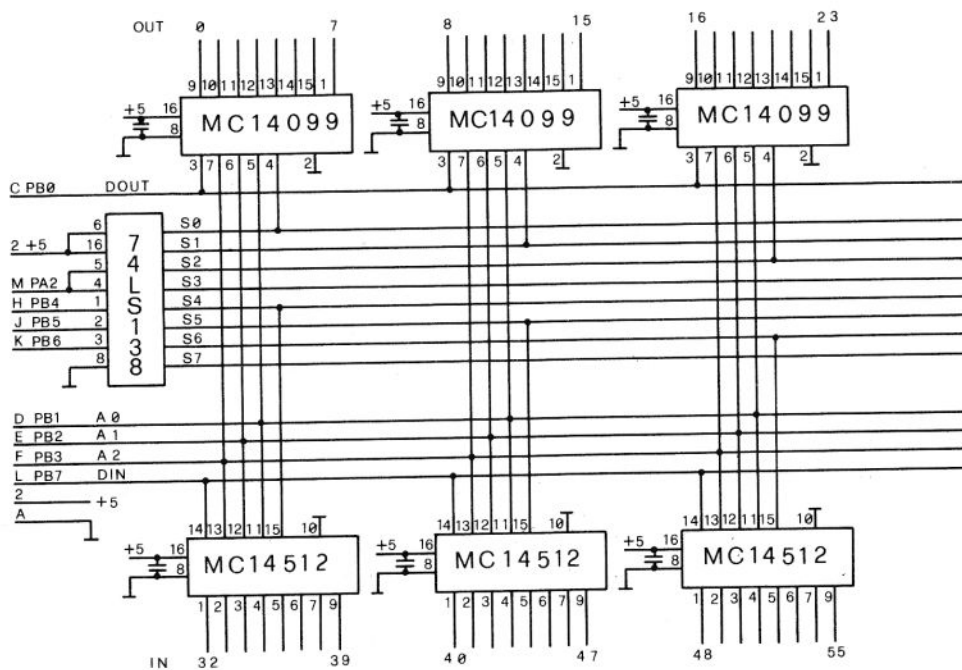



Fig.14 Schema elettrico della scheda per l'espansione seriale delle linee di I/O del C64

fine prematura. Lo stesso problema esiste per le uscite aggravato dal fatto che la bassa tensione e lapoca corrente erogabile dalle linee di output non sono nemmeno in grado di attivare rele' di piccola potenza, quindi e' necessaria quanto meno una robusta amplificazione di potenza affinche' gli ordini impartiti dal controllore possano essere eseguiti dall'apparecchiatura controllata. Quanto sopra non deve certo scoraggire ma piuttosto far riflettere sul fatto che ogni applicazione del computer nelcontrollo di apparecchiature si compone non solo di software (programma) ma non puo' ignorare le esigenze hardware (elettriche) dal momento che i valori logici sui quali lavora il nostro raffinatissimo programma derivano pur sempre da grandezze elettriche (correnti o tensioni) con le quali bisogna prima o poi fare i conti. L'elettronica, cacciata dall'informatica dalla porta, rientra dalla finestra e si fa sentire in tutto il suo giusto peso.

IL CIRCUITO ELETTRICO

Aumentare il numero delle linee di I/O del C64 puo' essere realizzate in molti modi, ad esempio aggiungendo sul Bus di espansione altri chip di I/O tipo 6521 o 6522 (rispettivamente detti PIA e VIA). Tenendo pero' conto delle altre esigenze elettriche accennate prima, una scelta ottimale e' costituita da chip di I/O seriali, rispettivamente l'MC14099B per le uscite e l'MC14512B per gli ingressi, entrambi molto economici e di facile reperibilita' dovunque. Il primo e' costuito da 8 elementi di memoria ad un bit (flip-flop) selezionabili singolarmente mediante gli ingressi A0-A1-A2 e tutti con l'ingresso in comune. In pratica, osservandone il pin-out, per impostare un certo valore logico (1 o 0) in un'uscita e' sufficiente presentarlo sul pin 3 (DATA) insieme all'indirizzo sui pin A e quindi dare un impulso negativo-positivo sul pin 4 (WRITE DISABLE). L'MC14512B puo' invece

Tabella 1 : USER-PORT del C64

PIN	TIPO	USO
A	GND	massa interna
B	FLAG2	
C	PB0	output dato
D	PB1	A0 indirizzo basso linea
E	PB2	A1 " medio "
F	PB3	A2 " alto "
H	PB4	A3 " basso chip
J	PB5	A4 " medio "
K	PB6	A5 " alto "
L	PB7	input dato
M	PA2	abilitazione chip
2	+5	alimentazione interna

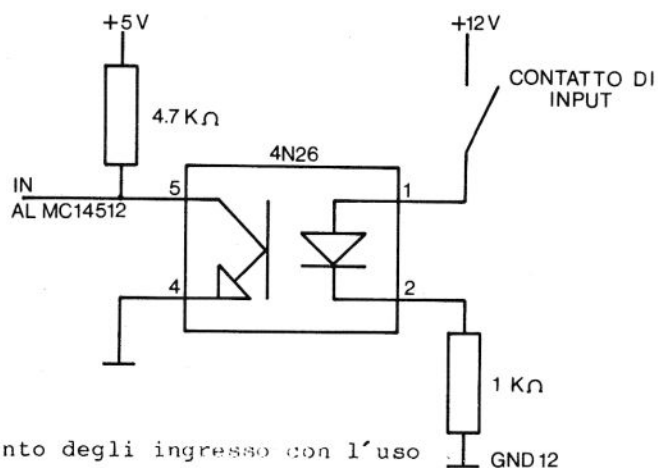



Fig.15 Disaccoppiamento degli ingresso con l'uso del fotoaccoppiatore 4N26

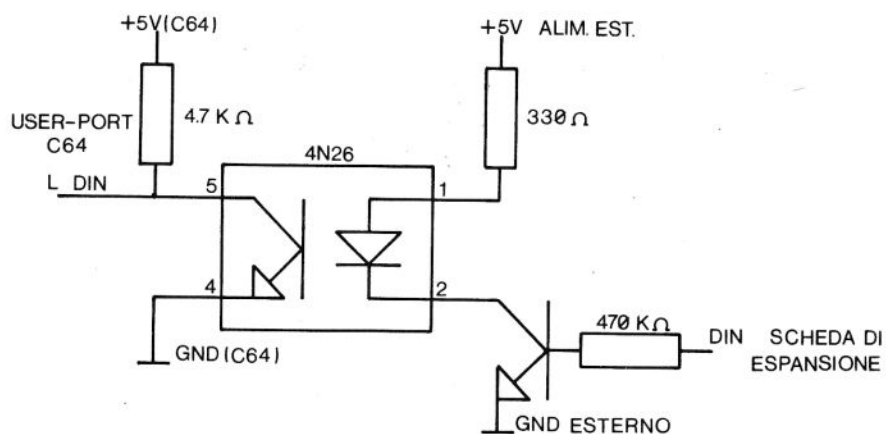



Fig.16 Disaccoppiamento completo della scheda di espansione seriale per mezzo di nove fotoaccoppiatori 4N26

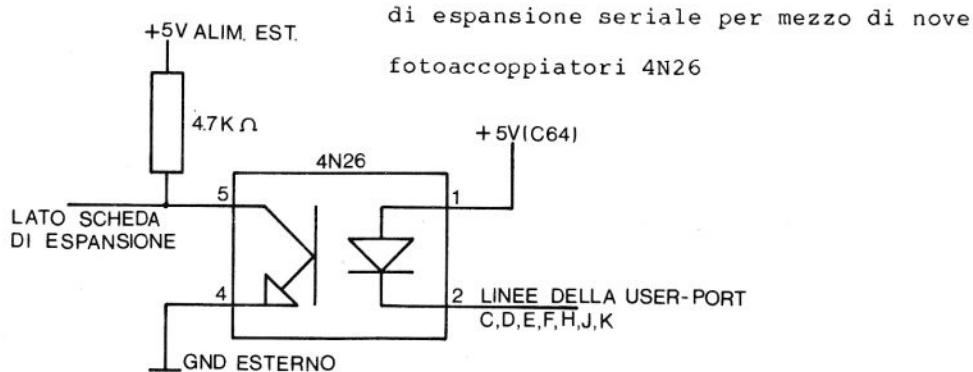
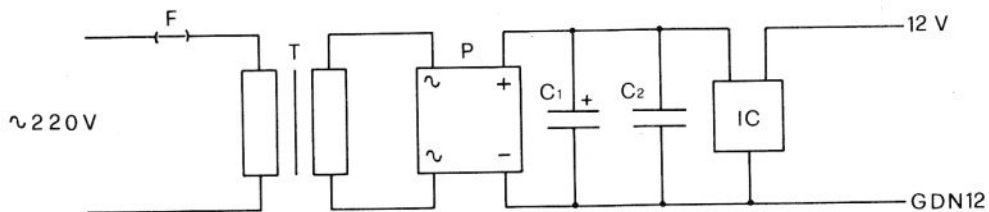



Fig.17 Alimentatore ausiliario necessario per

il disaccoppiamento ed il pilotaggio dei

carichi esterni

F = FUSIBILE 0,2 A
T = TRASFORMATORE
220/12 30 VA
P = RADDRIZZATORE A
PONTE 2 A 50 V
C1= 2200 microF. 25 Vl
C2= 0.1 microF.
IC= REGOLATORE DI
TENSIONE 7812

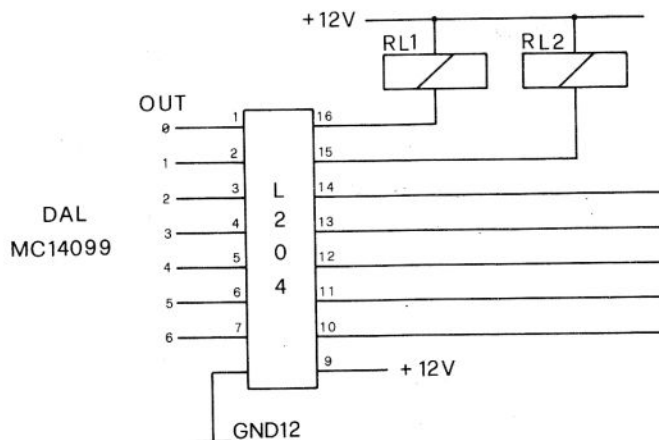



Fig.18 Pilotaggio delle uscite per mezzo di rele'

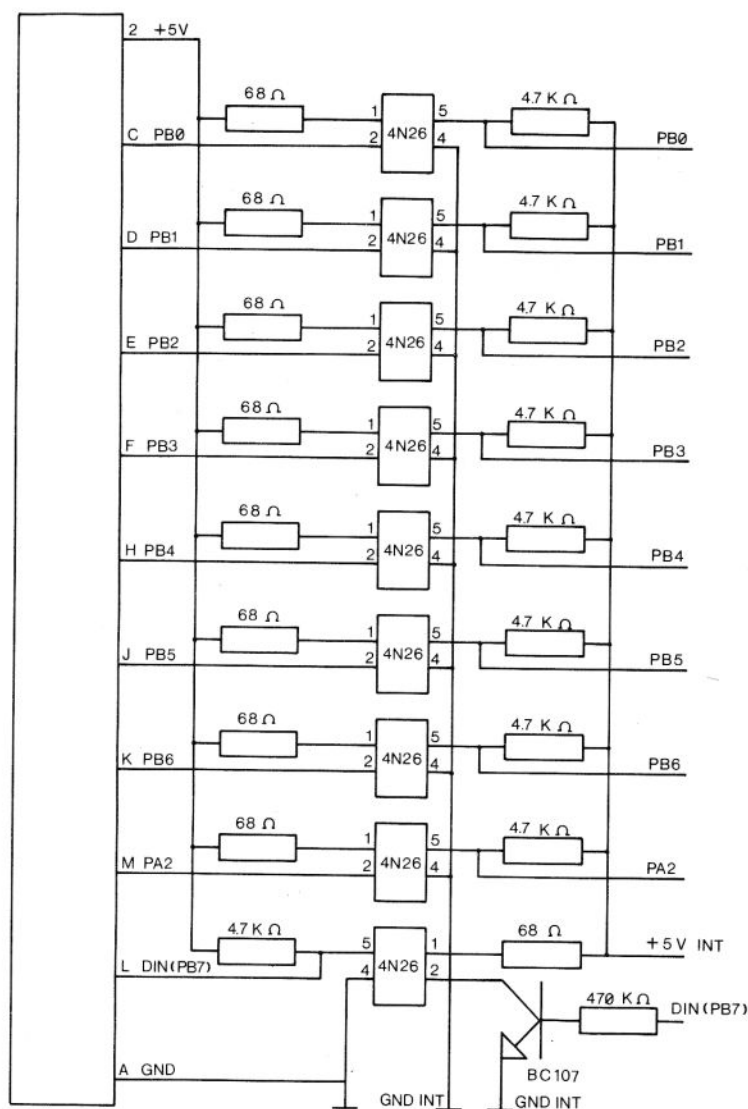
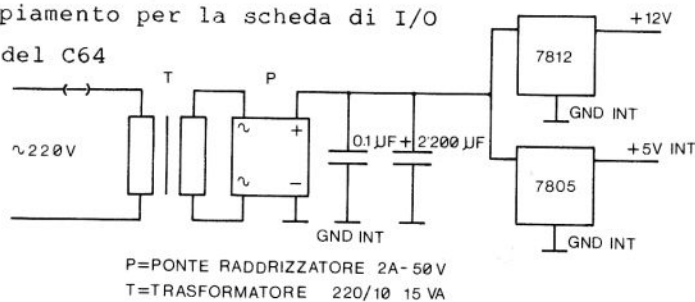



Fig.19 Schema completo della sezione di disaccoppiamento per la scheda di I/O seriale del C64

T
P
∿220V
0.1 μF
2200 μF
7812
+12V
7805
+5V INT
GND INT

P=PONTE RADDRIZZATORE 2A-50V
T=TRASFORMATORE 220/10 15 VA

essere pensato come un selettore a 8 posizioni comandato dai pin di indirizzo A0-A1-A2 e abilitabile mediante un livello zero sul pin 15 (DISABLE). Praticamente l'uscita Z rispecchiera' lo stato dell'ingresso il cui numero e' impostato sugli ingressi A per tutto il tempo che il pin DISABLE viene tenuto a zero. Mediante un integrato SN74LS138, decoder 3 a 8, e' possibile con solo tre linee, provenienti dal computer, abilitare singolarmente ben 8 Chip di I/O per un totale di 32 linee. La scelta di quanti input e quanti output dipendera' evidentemente dal tipo di applicazione da affrontare di volta in volta. Tutta la gestione puo' dunque avvienire tramite la USER-PORT del C64, ma qualunque computer dotato di almeno nove linee di I/O potra', con il software adatto, utilizzare lo stesso hardware con identici risultati. In tabella 1 e' specificato l'uso delle singole linee della USER-PORT del C64 per il controllo del circuito di input/output seriale.

Se non esistono problemi di distanza e di disturbi tutto potrebbe limitarsi allo schema indicato in Figura 14 diversamente converra' realizzare anche una completa separazione galvanica tra il computer ed il circuito di ingresso/uscita vero e proprio tramite disaccoppiatori ottici tipo 4N26 di cui si indica il pin-out e la struttura interna in figura 1. Praticamente essi sono costituiti da un diodo led accoppiato otticamente con un fototransistore in modo da poter trasferire un livello logico tra due circuiti dotati di masse separate. L'isolamento supera abbondantemente il Kilovolt e dovrebbe essere sufficiente anche nei casi piu' critici ad assicurare lunga e felice vita al computer ed al suo possessore. In questo caso pero' il circuito non potra' piu' usufruire dell'alimentazione del C64 e sara' dunque necessario realizzare un alimentatore separato in grado di erogare almeno una corrente di 0.5 Ampere alla tensione di 5 Volt come indicato nello schema

completo. Gli ingressi saranno attivabili mediante la chiusura (livello 1 in ingresso) o l'apertura di un semplice contatto meccanico capace di condurre qualche centesimo di Ampere, come un microinterruttore od un rele'.

Affinche' le uscite siano in grado di pilotare almeno la bobina di eccitazione di un rele' di media potenza e' indispensabile amplificarne il segnale opportunamente. Una buona soluzione e' rappresentata dall'uso dell'integrato L204 contenente sette transistori in configurazione Darlington separati e con uscita a collettore aperto in grado di sopportare sino a 30 V ed di assorbire senza problemi qualche centinaio di mA. Il pin-out riportato in figura mette anche in evidenza la presenza di un diodo accoppiato a ciascuna uscita, molto comodo per il pilotaggio di carichi induttivi quali le bobine di eccitazione dei rele'. Il fatto che ciscuno di questi integrati possa gestire solo 7 uscite e non 8 puo' rappresentare una piccola scomodita' ma non certamente un problema. Dal momento che in commercio esistono ottimi rele' eccitabili con 12 Volt e circa 50 mA sulla bobina, e che possono anche essere montati direttamente su circuito stampato, il gioco e' fatto. Tutta l'espansione di I/O puo' venire dunque realizzata su due basette di vetronite forata (meglio circuiti stampati), una contenente i disaccoppiatori ottici che andra' collegata direttamente alla USER-PORT del C64 ed una seconda che dovra' stare il piu' vicino possibile all'apparecchiatura da controllare, comprendente l'alimentatore ausiliario, i circuiti di decodifica e la sezione di potenza. Poiche' la comunicazione tra le due schede avviene in corrente (0 - 10 mA) senza connessione di massa non vi sono problemi di disturbi ne' pericoli di sorta per il computer ed il suo operatore. La distanza potra' raggiungere diverse decine di metri dipendendo in teoria solo dalla sezione dei fili usati. In

pratica oltre i 20-30 metri le capacita' tra i fili di collegamento potrebbero rallentare notevolmente i fronti di salita e di discesa dei segnali in transito rendendo problematica la comunicazione. Adottando circuiti trigger all'arrivo di ciascuna linea e rallentando la velocita' di comunicazione si potrebbe ovviare a questi inconvenienti complicando pero' notevolmente lo schema elettrico. Comunque il circuito presentato e' in grado di risolvere i problemi accennati all'inizio e di allargare notevolmente le possibilita' di applicazione del computer nel controllo di apparecchiature esterne sia in campo hobbistico che professionale

IL SOFTWARE

La gestione della comunicazione seriale nei due sensi e' realizzata, per evidenti motivi di velocita', in linguaggio macchina ed e' costituta da due routines, una di input ed una di output, richiamabili direttamente da Basic con l'istruzione SYS. Il numero della linea sulla quale operare dovra' trovarsi prima di ogni chiamata nella variabile riservata LL%. Dopo ogni input il valore letto (zero o uno) si trovera' direttamente nell'ultima variabile chiamata prima dell'istruzione SYS, che dovra' essere di tipo integer (%). Analogamente nel caso di output il valore voluto (0-1) dovra' trovarsi al momento del SYS nell'ultima variabile integer richiamata. Sono ammessi anche array purche' di tipo integer. Per chi volesse apportarvi modifiche e' riportato anche il listato Assembler dell'intera gestione. La routine del sistema operativo richiamata all'indirizzo $B0E7 si rivela molto utile in quanto restituisce nelle locazioni $47-$48 l'indirizzo di memoria nel quale si trova la variabile il cui nome e' stato posto in $45-$46.

Il programma Basic permette tramite i valori DATA di caricare in memoria le routines, spostando opportunamente il puntatore al top

della memoria, e nello stesso tempo fornisce un esempio di come sia possibile leggere il valore delle linee di input e scrivere in quelle di output dell'interfaccia seriale.

```
1 rem     ****************************
2 rem     *  gestione input/output   *
3 rem     *     seriale su C64       *
5 rem     ****************************
6 rem
8 rem   inizio programma di prova
9 rem
10 gosub1000:rem inizializzazione
15 rem
16 rem in ll% il numero della linea
17 rem in qualsiasi variabile integer (%)
18 rem il valore 0 oppure 1
19 rem
20 fork=0to7:rem ripete per otto linee
25 rem legge in v% lo stato della linea
26 rem del port 0
30 ll%=k:v%=v%:sys(40704):rem leggi
35 rem v% vale 0 oppure 1
40 printv%;
45 rem scrive v% nella linea di output
46 rem del port 5 (5*8=40)
50 ll%=k+40:v%=v%:sys(40707):rem scrivi
60 next
70 print:goto20:rem salta riga e da capo
998 end
999 rem
1000 rem programma in linguaggio macchina
1001 rem da porre in ram a partire da $9F00
1002 rem
1003 data  76 , 6 , 159 , 76 , 13 , 159
1005 data  169 , 0 , 72 , 169
1010 data  126 , 208 , 5 , 169 , 255 , 72 , 169 , 127 , 141 , 3
1020 data  221 , 141 , 1 , 221 , 169 , 4 , 13 , 2 , 221 , 141
1030 data  2 , 221 , 169 , 4 , 13 , 0 , 221 , 141 , 0 , 221
1040 data  165 , 71 , 72 , 165 , 72 , 72 , 169 , 204 , 133 , 69
1050 data  133 , 70 , 32 , 231 , 176 , 160 , 1 , 177 , 71 , 133
1060 data  251 , 104 , 133 , 72 , 104 , 133 , 71 , 104 , 240 , 24
1070 data  177 , 71 , 74 , 38 , 251 , 165 , 251 , 141 , 1 , 221
1080 data  173 , 0 , 221 , 41 , 251 , 141 , 0 , 221 , 9 , 4
1090 data  141 , 0 , 221 , 96 , 165 , 251 , 10 , 141 , 1 , 221
1100 data  173 , 0 , 221 , 41 , 251 , 141 , 0 , 221 , 173 , 1
1110 data  221 , 41 , 128 , 10 , 42 , 145 , 71 , 173 , 0 , 221
1120 data  208 , 222
1122 rem
1200 zz=9*16↑3+15*16↑2:rem $9F00
1205 rem divide parte alta e bassa
1210 zh=int(zz/256):zl=zz-zh*256
1215 rem sposta il top della memoria
1220 poke51,zl:poke52,zh:poke55,zl:poke56,zh
1225 rem scrive le routines in ram
1230 fork=zztozz+121:reada:pokek,a:next
1240 ll%=0
1250 return
1255 rem inizializzazione terminata
```

PRG11. Programma per la gestione
della scheda di input/output
seriale per il C64

INTERFACCIA ANALOGICA/DIGITALE

INTRODUZIONE

Uno degli aspetti piu' affascinanti dell' uso del computer e certamente quello destinato ad incidere sempre di piu' nel modo di vivere e di lavorare dell' uomo, e' rappresentato dalla possibilita' di realizzare sistemi automatici di controllo economici ed efficaci. Per Sistema Fisico si puo' intendere, ad esempio, un processo automatico di lavorazione ma, in senso piu' lato, anche l' ambiente in cui l' uomo si trova a vivere o a lavorare. Il computer puo' essere usato sia per il controllo della temperatura e dell' umidita' di una cella frigorifera che per il condizionamento di ambienti pubblici o domestici, cosi' come per la dosatura di componenti per miscele, con notevoli caratteristiche di precisione e di affidabilita'. La differenza sostanziale rispetto ai metodi di controllo classici e' rappresentata dal fatto che il sistema computerizzato e' in grado di adattare le sue modalita' di controllo alle differenti situazioni che si possono presentare secondo un programma di lavoro che puo' anche essere molto sofisticato senza bisogno di continui interventi da parte dell' operatore. Con riferimento all' esempio di condizionamento ambientale sara', ad esempio, possibile avere condizioni di temperatura diverse in ambienti diversi a seconda delle ore, delle condizioni atmosferiche esterne e del giorno della settimana con evidenti vantaggi sia dal punto di vista dell' economia di gestione che da quello di una migliore abitabilita'. Per sistema fisico si

intende un qualunque processo il cui stato e' completamente definito da una o piu' grandezze fisiche (temperatura, umidita', livello, ecc,) e sul quale e' possibile intervenire tramite una o piu' grandezze di controllo o attuatori (riscaldatore, compressore, valvole di carico e scarico, ecc.). Il controllore deve poter rilevare lo stato del sistema controllato con frequenza compatibile con le costanti di tempo caratteristiche ed intervenire sugli elementi di controllo (attuatori) affinche' tutto possa svolgersi secondo le modalita' stabilite nel programma di lavoro. Si consideri, come ulteriore esempio, un forno per cicli termici su determinati materiali nel quale la temperatura non debba rimenere costante ma variare in funzione del tempo in modo prestabilito. L' uso di un cronometro e di un buon termostato regolabile potrebbero risolvere il problema obbligando pero' l' operatore ad una continua presenza con costi elevati di gestione e possibilta' di errori. Sostituendo l' operatore, come sistema di elaborazione, con un computer si otterra' un controllo piu' assiduo della temperatura ed una migliore ripetibilta' del ciclo stesso. Quanto sopra vale a maggior ragione quando i parametri caratteristici del processo siano piu' di uno, ad esempio temperatura, pressione ed umidita', e la funzione di controllo sia piuttosto complessa come nel caso delle celle di essiccazione. Vi sono inoltre sistemi fisici sui quali non si vuole o non si puo' intervenire per il controllo dei parametri caratteristici, allora il computer puo' rivelarsi ancora prezioso come "DATA LOGGER" cioe' come strumento in grado di registrarne l' andamento nel tempo a fini statistici o di documentazione. La differenza fondamentale rispetto ai classici registratori su carta e' rappresentata dal fatto che i dati vengono memorizzati in questo caso in forma numerica con grande precisione e possono essere riletti dal computer per successive elaborazioni,

per analisi comparative o per la produzione di grafici. Un caso tipico potrebbe essere la registrazione, ad intervalli di tempo prefissati, dei valori di temperatura, umidita' e pressione atmosferica in aree adibite a particolari coltivazioni onde poterli correlare, in un secondo tempo, con la produzione ottenuta, o per la determinazione in tempo reale di possibili situazioni di rischio per le colture stesse. Gli esempi di possibile uso di un sistema di controllo computerizzato potrebbero continuare all' infinito abbracciando ogni settore dell' attivita' umana, ma ora conviene esaminare come esso possa essere realizzato ed in qual modo sia possibile programmarne il funzionamento secondo le modalita' richieste dal particolare sistema fisico da controllare.

L' ACQUISIZIONE

Come gia' accennato, le grandezze fisiche (pressione, temperatura ,livello, ecc.) per essere rilevabili dal computer dovranno essere trasformate dapprima in grandezze elettriche (tensione o corrente) ad esse proporzionali. A tal fine esiste in commercio una grande varieta' di dispositivi, detti TRASDUTTORI, che, con diverse caratteristiche e classi di precisione, sono in grado di di effettuare questa prima conversione. Nella tabella allegata sono indicati alcuni esempi relativi alle grandezza fisiche piu' comuni. Molto in uso e' la conversione in corrente 4-20 mA al fine di minimizzare l' influenza dei disturbi elettrici nel caso che il trasduttore debba trovarsi distante dal sistema di rilevazione vero e proprio. Cio' significa che al computer arrivera' una corrente compresa tra 4 mA e 20 mA corrispondenti ai limiti inferiore e superiore del campo di misura del trasduttore. Molto spesso gran parte del costo e delle caratteristiche dell' intero sistema di controllo dipendono da questa prima conversione, percio' la scelta

del tipo di trasduttore andra' fatta con grande cura tenendo conto delle specifiche di precisione e di scala fornite dal costruttore. Sara' ad esempio perfettamente inutile dimensionare un controllo di temperatura al decimo di grado quando la precisione del trasduttore sia dell' ordine del grado centigrado. Una volta che la grandezza fisica e' stata convertita in grandezza elettrica nasce, come noto, un altro problema. Infatti il computer e' una macchina digitale, cioe' in grado di riconoscere non il valore vero di tensione o di corrente ma soltanto se esso sia superiore ad una certa soglia (valore 1) o inferiore (valore 0) e quindi non di apprezzare ciascuno degli infiniti valori che la grandezza puo' assumere data la sua natura analogica. E' dunque necessario convertire quest' ultima nel corrispondente numero binario che il computer e' in grado di elaborare. Si rende dunque indispensabile l' adozione di un circuito in grado di svolgere questa ulteriore funzione, chiamato appunto Convertitore analogico/digitale. La scelta e' caduta sul semplice ed economico MC14433 della Motorola di reperibilta' molto facile essendo utilizzato in molti voltmetri digitali e strumenti di misura. In appendice ne sono indicate le caratteristiche principali e le modalita' di impiego desunte dalla documentazione fornita dal costruttore. Dal momento che le grandezze da controllare potrebbero essere piu' di una e' certamente utile anteporvi un multiplexer analogico tipo MC14051 dotato di 8 ingressi direttamente selezionabili tramite le linee del computer. In pratica lo si puo' pensare come un selettore a otto ingressi ed una uscita che si collega di volta all'ingresso il cui numero e' specificato su tre apposite linee di controllo. Attenzione a non confonderlo con i multiplexer digitali i quali trasferiscono alla loro uscita non il valore esatto della tensione di ingresso ma solo il valore logico. Con lo schema proposto e'

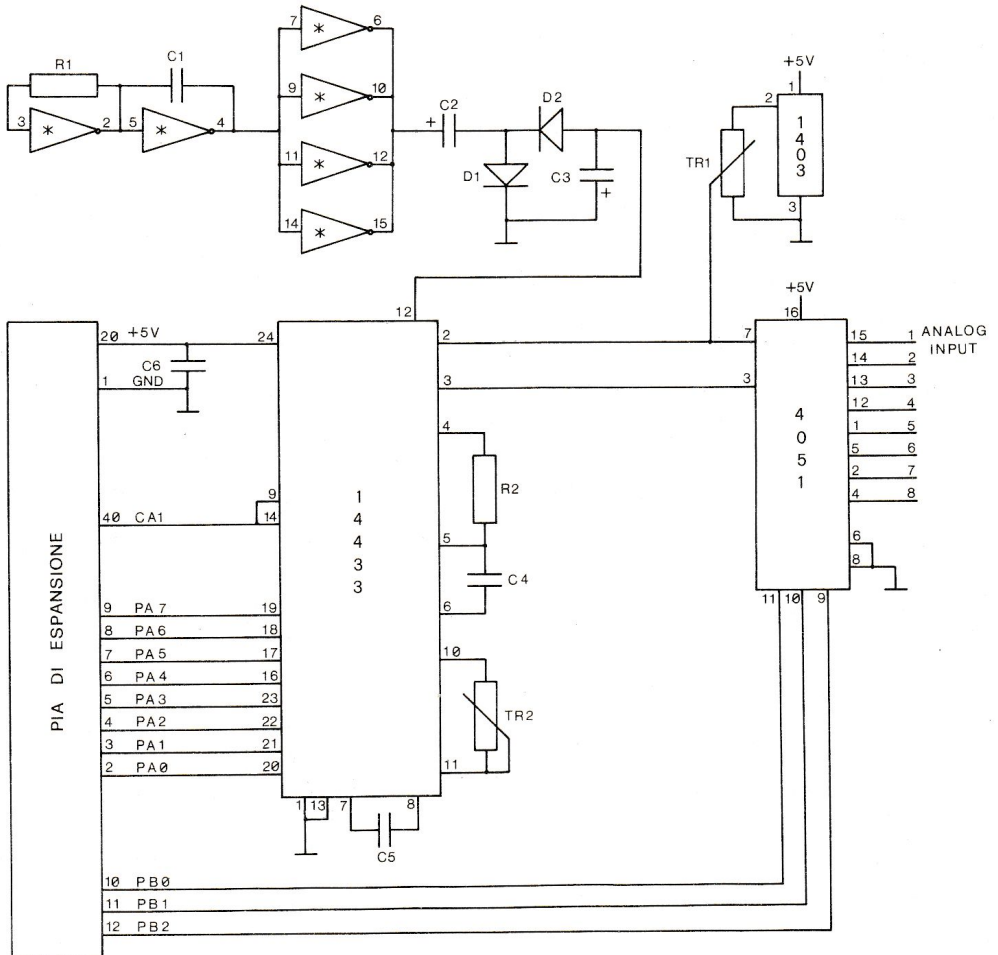
dunque possibile realizzare un sistema di acquisizione per otto grandezze analogiche con in piu' la possibilita' di pilotare ben sette attuatori tramite le restanti linee del chip di I/O 6521 adottato per il pilotaggio del covertitore.

IL SISTEMA DI ELABORAZIONE

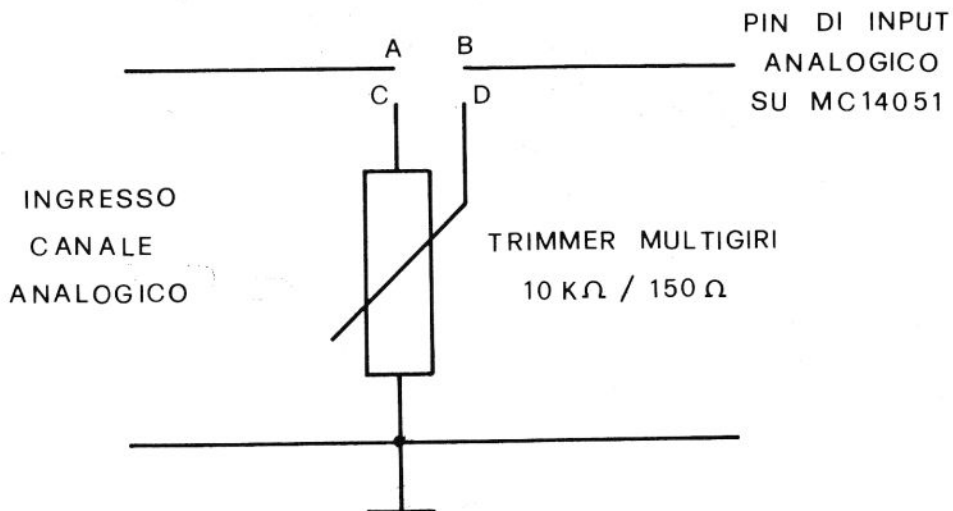
L'elemento che deve acquisire ed elaborare i dati provenienti dall'esterno e' il C64 che mantiene inalterate tutte le sue capacita' elaborative dal momento che il circuito di interfaccia occupa una piccola zona di memoria lasciata appositamente libera nella parte dedicata ai chip di I/O e cioe' da $DF84 a $DF88. La gestione vera e propria del convertitore potrebbe essere scritta direttamente in linguaggio Basic ma cio' rischierebbe di rallentare ulteriormente la velocita' di elaborazione per cui e' preferibile scriverla in linguaggio macchina e caricarla all'inizio del lavoro nella parte superiore della memoria RAM disponibile, da $9F00 a $9FA0, in modo da poterla richiamare quando necessario senza ulteriori problemi. Dunque, una volta caricata la routine in linguaggio macchina e programmate opportunamente le porte del chip di I/O 6521 (PORT A in input e PORT B in output), ciascuno potra' scrivere normalmente il proprio programma applicativo senza preoccuparsi ulteriormente della parte Hardware aggiuntiva. Se le rilevazioni dovranno avvenire su diverse linee di ingresso sara' necessario eseguire una conversione a vuoto dopo aver impostato il nuovo numero sul multiplexer per essere certi che il risultato della conversione sia effettivamente relativo al nuovo ingresso analogico e non al precedente o peggio ad un miscuglio dei due. Con questa piccola precauzione i dati letti rispecchieranno certamente il valore del segnale elettrico presente sull'ingresso selezionamto in quel momento.

* MC4011

PIN1 +5V

PIN8 GND

ELENCO COMPONENTI

C1=0.02 microF.	R1 =470 KOHM 1/4 WATT
C2=50 microF. 25 VL	R2 =470 KOHM 1/4 WATT
C3=50 microF. 25 VL	TR1=20 KOHM MULTIGIRI
C4=0.1 microF.	TR2=500 KOHM MULTIGIRI
C5=0.1 microF.	
C6=0.1 microF.	

Fig.20 Schema della scheda di interfaccia Analogico/Digitale con il convertitore MC14433

CONNESSIONI POSSIBILI

A - B TRIMMER ESCLUSO. INGRESSO DIRETTO PER TENSIONI INFERIORI A 2 VOLT.

A - C INGRESSO PER TENSIONI MAGGIORI DI 2 VOLT.
B - D REGOLARE IL TRIMMER SINO A PORTARE A 2 VOLT IL VALORE MASSIMO DI INGRESSO. TRASFORMAZIONE CORRENTE-TENSIONE PER I MAGGIORE DI 20 mA.

A - B INGRESSO PER CORRENTE 0-20 mA.
A - C REGOLARE IL TRIMMER PER AVERE 2 VOLT CON
C - D 20 mA IN INGRESSO. UTILIZZARE UN TRIMMER MULTIGIRI DI VALORE COMPRESO TRA 150 E 200 OHM.

Fig.21 Circuito di adattamento per gli ingressi analogici del multiplexer MC14051

LO SCHEMA ELETTRICO

Per interfacciare il convertitore e' stato necessario espandere le linee di I/O del C64 con un PIA 6521 connesso al port di espansione posteriore. Il collegamento ai bus del sistema e' quello gia' visto e l'indirizzamento sfrutta la linea I/O2 (indirizzi da $DF00 a $DFFF) e, per permettere ulteriori espansioni, anche le linee AB3 e AB7. I quattro registri del PIA si trovano dunque da $DF84 a $DF87 ma anche in tutti gli indirizzi compresi nella pagina di I/O2 caratterizzate da AB3 e AB7 contemporaneamente ad uno. Come gia' chiarito precedentemente, nel caso nascessero conflitti con altre schede di espansione bastera' scegliere la linea I/O1 oppure utilizzare altre linee del Bus degli indirizzi per evitare ogni problema. Il Port A del PIA e' usato come input dei risultati delle conversioni, una cifra alla volta, mentre le tre linee basse del PORT B selezionano il segnale analogico all'ingresso del multiplexer. Tramite CA1 il sistema e' in grado di sapere l'istante in cui e' finita una conversione ed e' quindi disponibile il risultato. Cinque linee del PORT B, CA2 e CB2 rimangono libere per il pilotaggio di carichi esterni quali rele', fotoaccoppiatori od altro. Negli esempi applicativo seguenti si vedra' piu' in dettaglio il loro uso e le modalita' di gestione delle linee di output. La parte relativa al convertitore e' praticamente desunta dalle note applicative relative al componente MC14433 e ben poco c'e' da aggiungere. Un particolare interessante e' la generazione di una tensione negativa di circa 3,5 Volt necessaria al funzionamento del sistema, ottenuta con un MC14011 usato sia come oscillatore che come buffer per un duplicatore di tensione a diodi. L' integrato MC1403 fornisce con buona stabilita' e precisione la tensione di riferimento per la conversione e tramite il trimmer TR1 e' possibile variarne il valore tra 2 V e 200 mVolt per adattarlo

ai diversi valori di fondo-scala richiesti. I trimmer di ingresso possono essere omessi o configurati come partitori di tensione per valori di input eccedenti i 2 Volt ed infine anche essere utilizzati per trasformare segnali in corrente tipo 4-20 mA in 0.4-2 Volt per l' ingresso del convertitore. Il valore della tensione di riferimento scelta rappresenta il fondo scala del convertitore. In tal senso andranno quindi interpretati i risultati: ad esempio il valore 1454 significa che in ingresso e' presente una tensione di 1.454 Volt se il riferimento e' a 2 Volt, diversamente dovra' essere interpretato come 0.727 Volt per un fondo scala di 1 Volt. I valori dei componenti indicati sono calcolati per un riferimento compreso tra 1 e 2 volt, nel caso fosse necessario ridurlo ulteriormente converra' riferirsi alla documentzione Motorola relativa al convertitore adottato. Una intera conversione dura circa 16400 cicli di clock del MC14433 per cui, avendo scelto una frequenza di 100 Khz tramite la resistenza Rc, si riesce ad acquisire sei dati al secondo e non oltre. Portando il clock a 400 Khz, massimo valore ammesso, le conversioni al secondo arriverebbero a circa 25 rendendosi pero' necessario ritoccare i valori degli altri componenti del circuito di acquisizione secondo le specifiche Motorola. Qualora pero' fosse necessaria una frequenza di campionamento elevata, l'adozione di un sistema di elaborazione programmato in Basic renderebbe problematica la gestione dei dati raccolti in tempo reale, per cui si puo' affermare che la velocita' scelta rappresenta un buon compromesso anche dal punto di vista dell'immunita' ai disturbi elettrici che purtroppo si trovano sempre sovrapposti al segnale da misurare. E' consigliabile realizzare l'intera interfaccia su circuito stampato prestando paricolare cura ai collegamenti di massa al fine di non penalizzare le buone caratteristiche di

```
1 rem       ****************************
2 rem       *   programma per l'uso    *
3 rem       *  del convertitore  a/d   *
4 rem       *     mc14433 sul c64      *
6 rem       ****************************
7 rem
8 rem       inizializzazione pia di i/o
9 rem       e routine di gestione
10 gosub10000
20 print"S":v%=0:rem pulizia schermo
30 print"linea =":rem numero linea da cui acquisire
40 getl$:ifl$=""then40
50 l=val(l$):ifl<0orl>7then50:rem compresa tra 0 e 7
60 poke(pi+2),l:rem output numero linea
65 v%=v%:sys40704:rem prima conversione di assestamento
70 v%=v%:sys40704:rem conversione vera e propria
80 print"linea ";l,"valore ";v%/100:rem output dato
90 getl$:ifl$=""then70:rem altra linea ?
100 ifval(l$)²0orval(l$)³7then90
110 goto50
9999 rem
10000 rem*****inizializzazione*******
10001 rem
10008 data  44 , 133 , 223 , 16 , 251 , 160 , 1 , 173 , 132 , 223
10010 data  170 , 41 , 16 , 240 , 248 , 138 , 41 , 15 , 153 , 153
10020 data  159 , 200 , 173 , 132 , 223
10025 data  170 , 41 , 32 , 240 , 248
10030 data  138 , 41 , 15 , 9 , 48 , 153 , 153 , 159 , 200 , 173
10040 data  132 , 223 , 170 , 41 , 64 , 240 , 248 , 138 , 41 , 15
10050 data  9 , 48 , 153 , 153 , 159 , 200 , 173 , 132 , 223 , 170
10060 data  41 , 128 , 240 , 248 , 138 , 41 , 15 , 9 , 48 , 153
10070 data  153 , 159 , 173 , 154 , 159
10075 data   72 , 162 , 43 , 41 , 4
10080 data  208 , 2 , 162 , 45 , 142 , 153 , 159 , 104 , 72 , 162
10090 data  48 , 41 , 8 , 208 , 2 , 162
10095 data  49 , 142 , 154 , 159
10100 data  104 , 162 , 50 , 41 , 9 , 201 , 1 , 208 , 3 , 142
10110 data  154 , 159 , 165 , 122 , 72 , 165 , 123 , 72 , 169 , 152
10120 data  133 , 122 , 169 , 159 , 133
10125 data  123 , 32 , 115 , 0 , 32
10130 data  243 , 188 , 104 , 133 , 123
10135 data  104 , 133 , 122 , 32 , 155
10140 data  188 , 160 , 1 , 165 , 101 , 145 , 71 , 136 , 165 , 100
10150 data  145 , 71 , 96
10198 rem     inizio routine $9f00
10200 zz=9*16↑3+15*16↑2
10210 zh=int(zz/256):zl=zz-zh*256
10219 rem     sposta il top della memoria
10220 poke51,zl:poke52,zh:poke55,zl:poke56,zh
10229 rem     carica la routine in ram
10230 fork=zztozz+152:reada:pokek,a:next
10239 rem     indirizzo chip di i/o $df84
10240 pi=13*16↑3+15*16↑2+8*16+4
10249 rem     inizializzazione porte
10250 pokepi+1,0:pokepi+3,0
10255 pokepi,0:pokepi+2,255
```

```
10256 pokepi+1,4:pokepi+3,4
10258 pokepi+2,0
10260 return
10270 rem  fine inizializzazione ora si puo' iniziare
```

PRG12 Esempio di utilizzo del
convertitore MC14433 con C64

```
1 rem            ************************
2 rem            * acquisizione dati e  *
3 rem            *  memorizzazione su   *
4 rem            * nastro ad intervalli *
5 rem            *  per commodore c64   *
7 rem            ************************
8 rem
10 gosub10000:rem inizializzazione
20 print"S":v%=0:rem pulizia schermo
30 input"ora hhmmss";ti$:rem ora di inizio
35 rem                         rilevazione
40 input"linea = 1 ";l :rem linea dalla quale
45 rem                        acquisire
50 pokepi+2,l
60 input"intervallo di acquisizione mm";mi
65 input"intervallo di acquisizione ss";si
70 i=(mi*60+si)*60:rem calcolo in sec/60
80 input"nome file";nf$:rem  file su cassetta
90 open1,1,1,nf$:rem apertura file sequenziale
100 v%=v%:sys40704:rem conversione in v%
115 tt$=ti$:tp=ti+i:rem salva ora
118 printti$,v%
120 print#1,ti$:rem registra l'ora
130 print#1,str$(v%):rem ed il valore rilevato
140 ti$=tt$:rem ripristina ora
150 ifti>tpthen100:rem se e' passato l'intervallo
160 getx$:ifx$=""then150
170 ifx$<>"f"then150:rem finito ?
180 print#1,"end":close1:rem chiude il file
190 end
10000 rem*****inizializzazione*******
10001 rem
10008 data  44 , 133 , 223 , 16 , 251 , 160 , 1 , 173 , 132 , 223
10010 data  170 , 41 , 16 , 240 , 248 , 138 , 41 , 15 , 153 , 153
10020 data  159 , 200 , 173 , 132 , 223
10025 data  170 , 41 , 32 , 240 , 248
10030 data  138 , 41 , 15 , 9 , 48 , 153 , 153 , 159 , 200 , 173
10040 data  132 , 223 , 170 , 41 , 64 , 240 , 248 , 138 , 41 , 15
10050 data  9 , 48 , 153 , 153 , 159 , 200 , 173 , 132 , 223 , 170
10060 data  41 , 128 , 240 , 248 , 138 , 41 , 15 , 9 , 48 , 153
10070 data  153 , 159 , 173 , 154 , 159
10075 data   72 , 162 , 43 , 41 , 4
10080 data  208 , 2 , 162 , 45 , 142 , 153 , 159 , 104 , 72 , 162
10090 data  48 , 41 , 8 , 208 , 2 , 162
10095 data  49 , 142 , 154 , 159
10100 data  104 , 162 , 50 , 41 , 9 , 201 , 1 , 208 , 3 , 142
10110 data  154 , 159 , 165 , 122 , 72 , 165 , 123 , 72 , 169 , 152
10120 data  133 , 122 , 169 , 159 , 133
10125 data  123 , 32 , 115 , 0 , 32
10130 data  243 , 188 , 104 , 133 , 123
10135 data  104 , 133 , 122 , 32 , 155
10140 data  188 , 160 , 1 , 165 , 101 , 145 , 71 , 136 , 165 , 100
10150 data  145 , 71 , 96
10200 zz=9*16↑3+15*16↑2
10210 zh=int(zz/256):zl=zz-zh*256
10220 poke51,zl:poke52,zh:poke55,zl:poke56,zh
10230 fork=zztozz+152:reada:pokek,a:next
```

```
10240 pi=13*16↑3+15*16↑2+8*16+4
10250 pokepi+1,0:pokepi+3,0
10255 pokepi,0:pokepi+2,255
10256 pokepi+1,4:pokepi+3,4
10258 pokepi+2,0
10260 return
20000 rem      *************************
20001 rem      *   rilettura dati da   *
20002 rem      *    nastro  e loro     *
20003 rem      *     elaborazione      *
20004 rem      *************************
20005 rem
20010 print"S":rem pulizia schermo
20020 print"qqqqqriavvolgi il nastro"
20025 getx$:ifx$=""then20025
20030 input"nome del file da caricare ";nf$
20035 input"qquanti dati";n:rem numero approssimativo
20036 rem                          dei dati memorizzati
20037 dimsb$(n,2):k=0
20040 open1,1,0,nf$
20050 input#1,sb$(k,1): rem rilettura ora
20055 ifsb$(k,1)="end"then20100
20060 input#1,sb$(k,2): rem rilettura valore
20080 k=k+1:goto20050
20100 close1
20110 print"i dati sono ";k
25000 rem   elaborazione dei dati
25010 rem   ed eventuale visualizzazione
25020 rem   su schermo o su plotter
25030 forn=0tok
25040 printsb$(n,1),sb$(n,2)
25050 next
25060 end
```

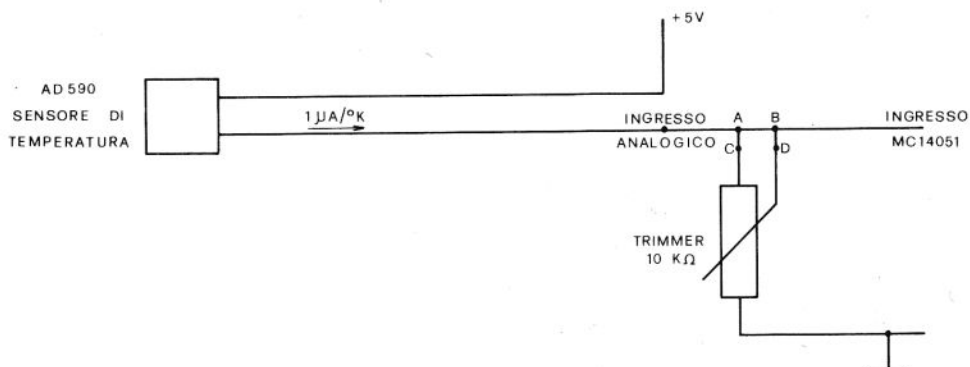
PRG13Uso del C64 come "DATA LOGGER"
per la memorizzazione dati e
successiva elaborazione

precisione e linearita' del convertitore adottato.

Un PRIMO ESEMPIO D'USO

Il programma PRG12 rappresenta il modo piu' semplice per usare l'interfaccia analogica/digitale. Una volta specificato il numero della linea sul video compaiono i valori che di volta in volta assume la tensione nell'ingresso scelto e di conseguenza la grandezza fisica ad essa proporzionale. Usando come trasduttore un AD590 posto, ad esempio, all'esterno dell'abitazione e' possibile avere in ogni istante il valore della temperatura, cosi' come la pressione e l'umidita' atmosferiche, disponendo degli opportuni sensori. Puo' trovare interessanti applicazioni anche per controllare l'andamento, ad esempio, di reazioni chimiche o altro in campo industriale.

UNA SECONDA APPLICAZIONE

Con PRG13 e' possibile non solo controllare l'andamento di una grandezza fisica ma anche memorizzarne i valori su nastro magnetico ad intervalli di tempo prefissati. Una cassetta di media capacita' puo' contenere un numero enorme (ben oltre mille) di rilevazioni permettendo dunque il funzionamento del C64 come DATA LOGGER anche per tempi prolungati. Premendo il tasto F si provoca il termine dell'operazione ed i dati rilevati rimangono a disposizione sul nastro per ogni successiva elaborazione od archiviazione. La seconda parte del programma, attivabile con RUN 20000,fornisce un esempio di come si possano rileggere i dati nella memoria interna del computer ad uso di un programma di gestione appositamente scritto. Modificando di poco il programma, aggiungendovi la gestione del multiplexer di ingresso, si potrebbero registrare i

TARATURA DEL SENSORE AD590

1) REGOLARE IL TRIMMER A META' CORSA (CIRCA 5 KOHM)
2) PORTARE IL SENSORE A 100 GRADI CENTIGRADI (373 KELVIN) E
   REGOLARE IL TRIMMER SINO AD AVERE 2 VOLT
   SULL'INGRESSO DEL MULTIPLEXER
3) CONTROLLARE E RITOCCARE IL TRIMMER PER AVERE
   1,47 VOLT CON IL SENSORE A 0 GRADI (273 KELVIN)

Fig.22 Collegamento e taratura del trasduttore

di temperatura AD590

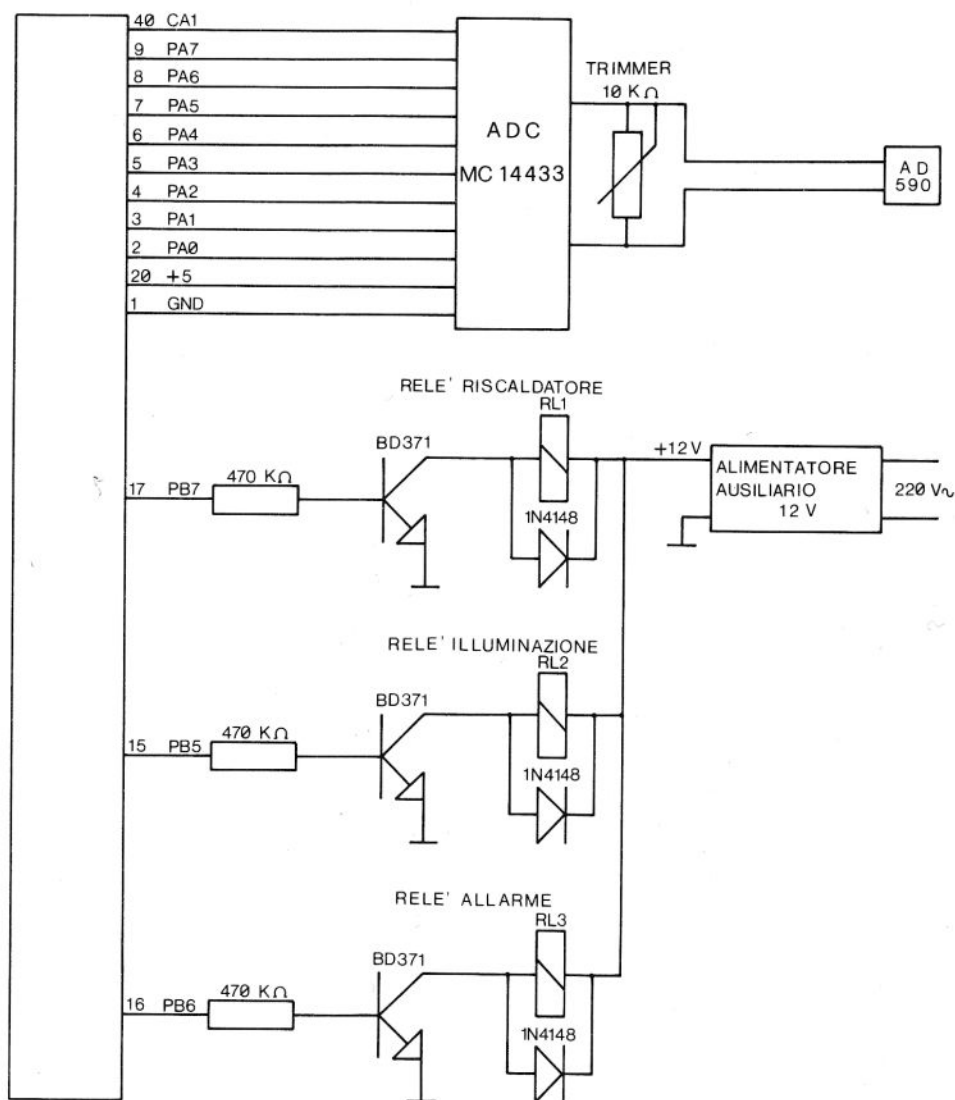



Fig.23 Schema elettrico della scheda per il controllo della temperatura dell'acqua e del tempo di illuminazione di un acquario con il C64

```
1 rem        ***************************
2 rem        *    controllo della      *
3 rem        *   temperatura e della   *
4 rem        *   illuminazione di un   *
5 rem        *    acquario con C64     *
7 rem        ***************************
8 rem*****uscita per riscldatore su pb7*
9 rem*****uscita di allarme su pb6 *****
10 rem****uscita illuminatore su PB5****11 gosub10000:rem
inizializzazione
12 rem
20 print"S":v%=0:rem pulizia schermo
30 input"linea l=";l:rem linea di input
40 ifl<0orl>7then30:rem compresa tra 0  e 7
50 input"set-point ";sp:rem valore da mantenere
59 rem   di quanto puo´ andare oltre
60 input"allarme superiore +";as
70 ls=sp+as
79 rem   di quanto puo´ stare sotto
80 input"allarme inferiore -";ai
90 li=sp-ai
93 rem   di quanto puo´ oscillare
94 rem       intorno al valore voluto
95 input"isteresi =";is
96 input"qilluminazione dalle hhmm ";lo
97 input"q    sino       alle  hhmm ";lf
98 input"q  ora   hhmmss ";ti$
99 rem inizio loop di controllo
100 v%=v%:sys40704:rem conversione a/d
110 v=(int((v%/5)-273)*100)/100
120 rem  ac=1 allora accende l´attuatore
125 rem  ac=0 allora spegne l´attuatore
130 ifv>(sp+is/2)then:ac=0:goto145
140 ifv<(sp-is/2)then:ac=1
144 rem  spegnimento attuatore pb7
145 ifac=0then:v$=str$(v):pokepi+2,(peek(pi+2)and(255-128))
149 rem  accensione attuatore pb7
150 ifac=1then:v$="r"+str$(v)+"R":pokepi+2,(peek(pi+2)or128)
155 al$=""
160 ifv>lsthen:al$="allarme superiore"
170 ifv<lithen:al$="allarme inferiore"
179 rem  spegnimento allarme pb6
180 ifal$=""then:pokepi+2,(peek(pi+2)and(255-64))
189 rem  accensione allarme pb6
190 ifal$<>""then:pokepi+2,(peek(pi+2)or64)
191 rem   controllo illuminazione*****
192 hm=int(val(ti$)/100)
193 rem     accende illuminazione
194 ifhm>=loandhm<lfthen:pokepi+2,(peek(pi+2)or32):ll$="raccesaR"
195 rem    spegne illuminazione
196
ifhm<loorhm>=lfthen:pokepi+2,(peek(pi+2)and(255-32)):ll$="spenta"
200 gosub20000:rem visualizzazione
201 rem
202 rem  variazione dati durante il funzionamento
203 rem "s" per set-point
204 rem "+" per allarme superiore
```

```
205 rem "-" per allarme inferiore
206 rem
210 getx$:ifx$=""then100
220 ifx$="s"then:input"set-point";sp:ls=sp+as:li=sp-ai:goto100
230 ifx$="+"then:input"allarme superiore =";as:ls=sp+as:goto100
240 ifx$="-"then:input"allarme inferiore =";ai:li=sp-ai:goto100
250 goto100
999 end
10000 rem*****inizializzazione*******
10001 rem
10008 data  44 , 133 , 223 , 16 , 251 , 160 , 1 , 173 , 132 , 223
10010 data  170 , 41 , 16 , 240 , 248 , 138 , 41 , 15 , 153 , 153
10020 data  159 , 200 , 173 , 132 , 223
10025 data  170 , 41 , 32 , 240 , 248
10030 data  138 , 41 , 15 , 9 , 48 , 153 , 153 , 159 , 200 , 173
10040 data  132 , 223 , 170 , 41 , 64 , 240 , 248 , 138 , 41 , 15
10050 data  9 , 48 , 153 , 153 , 159 , 200 , 173 , 132 , 223 , 170
10060 data  41 , 128 , 240 , 248 , 138 , 41 , 15 , 9 , 48 , 153
10070 data  153 , 159 , 173 , 154 , 159
10075 data   72 , 162 , 43 , 41 , 4
10080 data  208 , 2 , 162 , 45 , 142 , 153 , 159 , 104 , 72 , 162
10090 data  48 , 41 , 8 , 208 , 2 , 162
10095 data  49 , 142 , 154 , 159
10100 data  104 , 162 , 50 , 41 , 9 , 201 , 1 , 208 , 3 , 142
10110 data  154 , 159 , 165 , 122 , 72 , 165 , 123 , 72 , 169 , 152
10120 data  133 , 122 , 169 , 159 , 133
10125 data  123 , 32 , 115 , 0 , 32
10130 data  243 , 188 , 104 , 133 , 123
10135 data  104 , 133 , 122 , 32 , 155
10140 data  188 , 160 , 1 , 165 , 101 , 145 , 71 , 136 , 165 , 100
10150 data  145 , 71 , 96
10200 zz=9*16↑3+15*16↑2
10210 zh=int(zz/256):zl=zz-zh*256
10220 poke51,zl:poke52,zh:poke55,zl:poke56,zh
10230 fork=zztozz+152:reada:pokek,a:next
10240 pi=13*16↑3+15*16↑2+8*16+4
10250 pokepi+1,0:pokepi+3,0
10255 pokepi,0:pokepi+2,255
10256 pokepi+1,4:pokepi+3,4
10258 pokepi+2,0
10260 return
20000 rem routine di visualizzazione
20010 rem
20020 print"S":rem pulizia schermo
20025 print"        rsituazione alle ";ti$;"R"
20026 print"qqq"
20030 print"qqrset-point","temperatura","R"
20040 print"qqÜ";sp,"ÜÜ";v$,al$:rem output dati
20050 print"qqqqqqÜÜÜÜrilluminazione R";ll$
20060 fort=0to400:nextt
20070 return
```

PRG14 Gestione di un acquario con C64

dati provenienti da piu' sensori in ogni intervallo rendendo quindi possibile analisi complete dell'andamento di un certo fenomeno con correlazioni tra le diverse grandezze in gioco.

IL PROBLEMA DELL'ACQUARIO

Il programma PRG14 rappresenta un vero e proprio esempio di controllo di processo e puo' essere molto utile per chi si dedica all'allevamento dei pesci in acquario. Come noto la temperatura dell'acqua deve mantenersi entro limiti molto stretti ed il tempo di illumimazione deve essere controllato con precisione onde non recare danni ai piccoli amici acquatici. Con una sonda di temperatura tipo AD590 e tre rele' di piccola potenza e' possibile controllare i suddetti parametri con estrema precisione e senza distrazioni. E' previsto anche di poter fissare due limiti di sicurezza, uno inferiore ed uno superiore, per la temperatura al di fuori dei quali entra in funzione un allarme, un campanello od una piccola sirena, per segnalare che la salute degli abitanti dell'acquario sta correndo seri pericoli. Come solito il programma si presta a modifiche e personalizzazioni che ciascuno, in base all'esperienza, puo' apportare per affinare le funzioni dell'intero sistema di controllo computerizzato.

UN ESEMPIO APPLICATIVO

Al fine di chiarire ulteriormente le potenzialita' ed il tipo di uso dell'interfaccia di acquisizione si supponga di dover controllare singolarmente la temperatura in un appartamento composto da 6 locali ed inoltre si voglia anche gestire il funzionamento del bruciatore della caldaia centralizzata. In ogni stanza sara' necessario disporre un sensore tipo AD590 per la

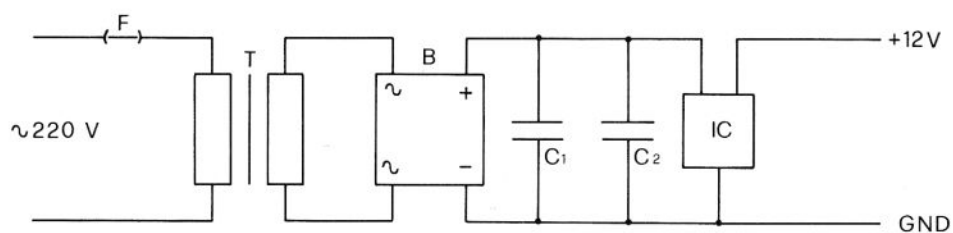



Fig.24 Alimentatore stabilizzato ausiliario
a 12 Volt

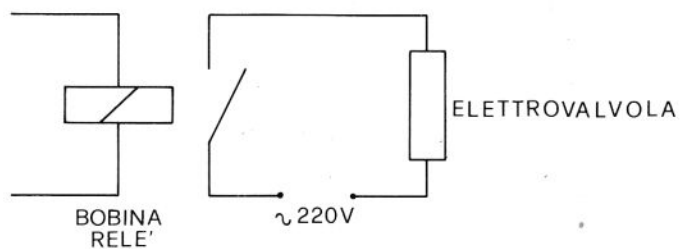



Fig.25 Connessione dei contatti dei rele' di
comando alle elettrovalvole di mandata
dell'acqua calda alle varie stanze

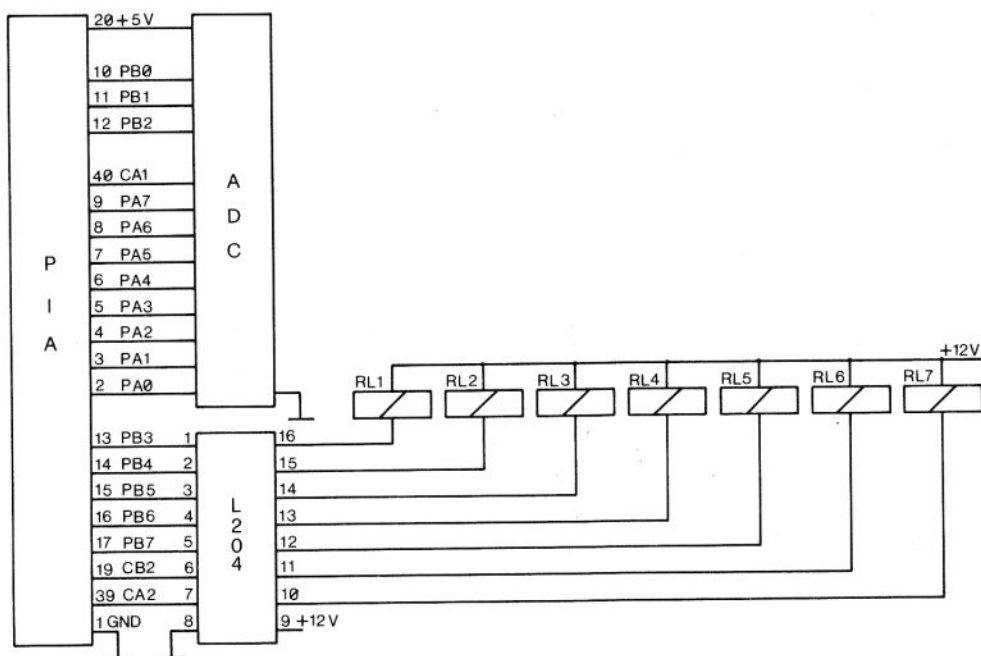



Fig.26 Circuito completo per il controllo della temperatura di un appartamento con il C64

```
1 rem    ******************************
2 rem    * programma per la gestione   *
3 rem    *  del riscaldamento di un    *
4 rem    * appartamento con il C64     *
6 rem    ******************************
7 rem
8 rem       inizializzazione pia di i/o
9 rem       e routine di gestione
10 gosub10000
20 print"S":dimv%(10):rem pulizia schermo
140 input"qqqqora hhmmss";ti$:print"S"
150 gosub11000:rem carica i nmi delle stanze
160 gosub12000:rem input delle temperature
170 rem                  di confronto
200 fors=0to6
205 pokepi+2,(peek(pi+2)and(128+64+32+16+8)ors)
210 v%=v%:sys40704:rem pre-conversione per
215 rem                   assestamento linea
218 rem
219 rem conversione e calcolo temperatura
220 v%=v%:sys40704:te(s)=(int(((v%/5)-273)*100))/100
230 next
240 gosub13000:rem gestione elettrovalvole
290 print"S"
295 rem ****visualizzazione delle temp.**
300 print"              situazione alle r";ti$:print:print
305 print"r";"stanza",,"temp","set-p";"R":print
310 fors=0to6:print"r";n$(s);"R",,te$(s),sp(s);"q":next
315 print:print:print"r";g$;"R"
318 rem***test per cambio set-point***
320 getx$:ifx$=""then200:rem no continua
330 ifx$<>"s"then200
340 gosub12000:rem variazione set-point
350 goto200:rem continua
9998 rem
9999 rem
10000 rem*****inizializzazione*******
10001 rem
10008 data  44 , 133 , 223 , 16 , 251 , 160 , 1 , 173 , 132 , 223
10010 data  170 , 41 , 16 , 240 , 248 , 138 , 41 , 15 , 153 , 153
10020 data  159 , 200 , 173 , 132 , 223
10025 data  170 , 41 , 32 , 240 , 248
10030 data  138 , 41 , 15 , 9 , 48 , 153 , 153 , 159 , 200 , 173
10040 data  132 , 223 , 170 , 41 , 64 , 240 , 248 , 138 , 41 , 15
10050 data  9 , 48 , 153 , 153 , 159 , 200 , 173 , 132 , 223 , 170
10060 data  41 , 128 , 240 , 248 , 138 , 41 , 15 , 9 , 48 , 153
10070 data  153 , 159 , 173 , 154 , 159
10075 data   72 , 162 , 43 , 41 , 4
10080 data  208 , 2 , 162 , 45 , 142 , 153 , 159 , 104 , 72 , 162
10090 data  48 , 41 , 8 , 208 , 2 , 162
10095 data  49 , 142 , 154 , 159
10100 data  104 , 162 , 50 , 41 , 9 , 201 , 1 , 208 , 3 , 142
10110 data  154 , 159 , 165 , 122 , 72 , 165 , 123 , 72 , 169 , 152
10120 data  133 , 122 , 169 , 159 , 133
10125 data  123 , 32 , 115 , 0 , 32
10130 data  243 , 188 , 104 , 133 , 123
10135 data  104 , 133 , 122 , 32 , 155
```

```
10140 data  188 , 160 , 1 , 165 , 101 , 145 , 71 , 136 , 165 , 100
10150 data  145 , 71 , 96
10198 rem     inizio routine $9F00
10200 zz=9*16↑3+15*16↑2
10210 zh=int(zz/256):zl=zz-zh*256
10219 rem     sposta il top della memoria
10220 poke51,zl:poke52,zh:poke55,zl:poke56,zh
10229 rem     carica la routine in ram
10230 fork=zztozz+152:reada:pokek,a:next
10239 rem     indirizzo chip di i/o $DF84
10240 pi=13*16↑3+15*16↑2+8*16+4
10249 rem     inizializzazione porte
10250 pokepi+1,0:pokepi+3,0
10254 pokepi,0:pokepi+2,255
10255 rem CA2 e CB2 a zero***********
10256 pokepi+1,(32+16+4):pokepi+3,(32+16+4)
10258 pokepi+2,0:rem gli altri out a zero
10260 retucn
10270 rem  fine inizializzazione ora si puo' iniziare
11000 rem*******nomi delle stanze***
11010 fors=0to6:readn$(s):next
11020 return
11030 data cucina,bagno,soggiorno,camera a,camera b,studio,caldaia
12000 rem*****input set-point********
12010 print"S":print
12020 fors=0to6
12030 print"temperatura in ";n$(s);"  "sp(s);
12035 t=0:inputt:ift=0thent=sp(s)
12040 sp(s)=t
12050 next
12060 return
13000 rem**attivazione elttrovalvole***
13010 n=8:g$=""
13020 fors=0to6
13030 ifte(s)<(sp(s)-1)then:c(s)=1:rem accendere
13040 ifte(s)>(sp(s)+1)then:c(s)=0:rem spegnere
13045 ifs³4then13080:rem casi particolari
13046 rem
13048 rem*attivazione o no dell'elettrovalvola
13050
ifc(s)=0then:te$(s)=str$(te(s)):pokepi+2,(peek(pi+2)and(255-n)):got
o13150
13060 te$(s)="r"+str$(te(s))+"R":pokepi+2,(peek(pi+2)orn):goto13150
13080 ifs=6then13100
13083 rem **********caso di ca2
13085
ifc(s)=0then:te$(s)=str$(te(s)):pokepi+1,(peek(pi+1)and(255-8)):got
o13150
13090 te$(s)="r"+str$(te(s))+"R":pokepi+1,(peek(pi+1)or8):goto13150
13099 rem **********caso di cb2
13100
ifc(s)=0then:te$(s)=str$(te(s)):pokepi+3,(peek(pi+3)and(255-8)):got
o13150
13110 te$(s)="r"+str$(te(s))+"R":pokepi+3,(peek(pi+3)or8):goto13150
13148 rem *******attenzione ai guasti !
13150 ifte(s)>98 then:te$(s)="rk.o.R":g$="guasto in "+n$(s)
13160 next
13170 return
```

PRG15 Gestione del riscaldamento di
un intero appartamento con il C64

rilevazione continua della temperatura e di una elettrovalvola con la quale poter comandare l'invio dell'acqua proveniente dalla caldaia centrale. Per quest'ultima si dovra' poter conoscere la temperatura dell'acqua e controllare l'accensione del bruciatore con una delle uscite del computer. Il programma, in base ai valori di temperatura impostati singolarmente per ogni locale (SET-POINT), si occupa di attivare o meno le elettrovalvole di controllo per ciascun radiatore e di mantenere l'acqua nella caldaia alla temperatura voluta. All'inizio vengono chiesti i set-point di partenza ed in ogni istante e' possibile modificarli singolarmente premendo il tasto S per adattarli alle diverse ore del giorno. Il programma fornito si presta pero' ad essere modificato al fine di ottenere funzioni molto interessanti. Ad esempio, dal momento che e' il computer ad attivare il bruciatore della caldaia, e' abbastanza facile rilevare il consumo di combustibile in un dato periodo semplicemente sommando i tempi di accensione e moltiplicando il risultato per il consumo orario rilevato sperimentalmente o fornito dal costruttore. Visto poi che il C64 e' dotato di un preciso orologio sarebbe anche possibile stabilire a priori diverse distribuzioni delle temperature nelle stanze in base alle ore del giorno e della notte e fare si che i set-point si modifichino automaticamente con il passare del tempo secondo le diverse esigenze. Chi fosse veramente curioso potrebbe anche sapere quanto del consumo globale di combustibile e' da imputare ad ogni stanza semplicemente sommando i tempi si apertura delle singole elettrovalvole di mandata dell'acqua di riscaldamento riuscendo in tal modo a scoprire eventuali difetti di isolamento termico negli infissi o nelle pareti di ciascun locale. Come si vede l'interfaccia presentata si presta ad applicazioni molto interessanti non solo in campo domestico ma anche industriale dal

momento che il sensore di temperatura ha l'uscita in corrente e puo' quindi essere posto anche a diverse decine di metri dal computer senza problemi di disturbi od altro, cosi' come gli attuatori comandati dai rele' di uscita. Il sensore di temperatura rappresenta certamente il trasduttore di ingresso piu' semplice ma a seconda delle esigenze di controllo qualunque altro tipo potra' essere gestito nello stesso modo visto sopra.

L'OROLOGIO HARDWARE
DEL COMMODORE C64

INTRODUZIONE

Praticamente tutti i computer oggi in commercio sono dotati di un orologio interno che permette la misura di intervalli di tempo durante l'esecuzione di un programma o semplicemente di sapere che ore sono in qualsiasi istante. Anche il Commodore C64 lo possiede, infatti basta impostare od interrogare le variabili riservate TIME$ o TIME per rendersene conto. Quest'ultima e' di tipo numerico e a sola lettura, cioe'non puo' essere scritta direttamente ma solo interrogata, e conta in sessantesimi di secondo senza suddivisioni in ore minuti e secondi. Puo' essere usata per misurare intervalli di tempo con discreta precisione. La variabile TIME$ invece costituisce il vero e proprio orologio del C64. Essa infatti puo' essere impostata ad una ora precisa, provocando in tal modo l'automatico aggiornamento di TIME, e interrogata in ogni momento successivo per conoscere l'ora esatta come un normale orologio digitale. Il tempo viene contato su 24 ore giornaliere con la risoluzione del secondo. Fin qui tutto bene, come racconta molto succintamente il manuale Commodore. Se pero'qualcuno si azzarda ad usare il computer per memorizzare su disco o peggio su nastro una serie di dati rilevati periodicamente da una apparecchiatura esterna o semplicemente compie frequenti accessi alle unita' periferiche, si accorge tristemente che l'orologio interno comincia a presentare un vistoso quanto inspiegabile ritardo. Tutto quindi va bene finche' non si usa la cassetta o il disco, diversamente la misura del tempo tramite l'orologio interno diventa assolutamente inaccettabile. Questo aspetto che puo' essere scambiato per un malfunzionamento legato ad un particolare C64 e' invece dovuto al

modo col quale esso gestisce l'orologio interno e quindi paradossalmente normale. Naturalmente i manuali Commodore ben si guardano dal sottolineare questa particolarita' probabilmente per stimolare la fantasia ed i nervi del programmatore. La spiegazione e' comunque molto semplice, infatti sessanta volte al secondo un Timer interno al C64 provoca una interruzione mascherabile (IRQ) alla CPU che esegue di conseguenza l'aggiornamento delle locazioni di memoria che contengono le ore, i minuti, i secondi ed i sessantesimi affinche' l'orologio sia sempre esatto. La temporizzazione deriva dal quarzo stesso che genera il clock del sistema ed e' quindi molto precisa. Purtroppo pero' quando il computer comunica con il disco o la cassetta queste interruzioni vengono ignorate (mascherate) per non perdere tempo prezioso e quindi il conteggio del tempo perde colpi. L'accesso a cassetta fa perdere diversi secondi mentre quello a disco pochi decimi. Naturalmente lavorando solo sulla memoria interna l'orologio mantiene la sua naturale precisione. I progettisti della Commodore hanno scelto di privilegiare il dialogo con le periferiche rispetto al conteggio del tempo e tutto sommato ben poco si puo' fare per porvi rimedio. Il C64 pero' contiene ben due precisi orologi hardware uno per ciscuno dei chip 6526 che lo equipaggiano e dunque perche' non sfruttarli.

L'OROLOGIO DEL 6526

Il C64 usa per dialogare con l'esterno due chip periferici 6526 appositamente realizzati che rappresentano una evoluzione del piu' noto 6522 (VIA) adottato sul buon VIC 20. All'interno di ciascuno di essi si trova, tra l'altro, un completo orologio con addirittura la possibilita' di impostare un allarme ad una ora precisa detto TIMER TOD. Si tratta di un contatore particolare per usi in tempo

reale. L'orologio copre 24 ore (AM/PM) con la risoluzione del decimo di secondo. E'organizzato su quattro registri: decimi di secondo, secondi, minuti e ore. L'indicazione AM/PM (mattina/pomeriggio) si trova nel MSB (Most Significant Bit cioe' bit piu' significativo) del registro delle ore per facilitare l'interpretazione dell'ora. Ogni registro contiene i valori in formato BCD (decimale codificato in binario) per una lettura piu' facile direttamente da Basic. L'orologio richede un segnale di ingresso a 60 o 50 Hz (programmabile) con livello TTL sul pin TOD per una misura accurata del tempo. Inoltre si ha a disposizione anche un allarme programmabile per la generazione di interruzioni alla CPU all'istante desiderato. I registri per l'impostazione dell'ora di allarme occupano gli stessi indirizzi di quelli dell'orologio vero e proprio ed il loro accesso e' governato dallo stato di un bit del registro di controllo. L'ora di allarme risiede in memorie a sola scrittura; qualsiasi operazione di lettura accedera' sempre ai registri di conteggio dell'orologio senza riguardo per lo stato del bit del registro di controllo.

Per impostare e leggere correttamente i registri del TOD si deve seguire una procedura ben precisa. Non appena si effettua una scrittura sul registro delle ore, il TOD si ferma e non riparte sino alla successiva scrittura del registro dei decimi di secondo. Cio' ne assicura la partenza sempre con il valore desiderato. Poiche durante un'operazione di lettura dell'ora puo' verificarsi un riporto tra i diversi registri e' prevista una funzione di memorizzazione per mantenere costanti i valori dei registri durante tutta la durata dell'operazione di lettura. Ad una lettura del registro delle ore, tutti gli altri vengono memorizzati in registri "latch" rimanendo in questo stato finche' non si verifica una scrittura sul registro dei decimi do secondo. Anche pero' durante

questo tempo il TOD continua ad aggiornare il tempo correttamente. L'indicazione delle ore e' compresa tra 01 e 12 e cio' crea qualche problema nella conversione su 24 ore perche' ad esempio appena dopo le 11 AM (le nostre 11 del mattino) vengono le 12 PM (il pomeriggio inizia a mezzogiorno!), mentre alle 11 PM (le nostre 23) fanno seguito le 12 AM (24). Con un po' di attenzione pero' e' possibile trasformare l'indicazione americana in quella europea a 24 ore giornaliere. La precisione e' data da quella della frequenza di rete ed e' normalmente molto buona, mentre la risoluzione si ferma ai decimi di secondo. Un tale orologio, essendo completamente hardware, non risentira' delle operazioni software per cui manterra' la sua precisione indipendentemente dal programma in esecuzione. Naturalmente dopo ogni accensione sara' necessario impostare di nuovo l'ora e l'eventuale ora di allarme, dopo di che l'orologio continuera' ad aggiornarsi ogni decimo di secondo senza problemi.

LA GESTIONE DELL'OROLOGIO

Di seguito sono riportate le routine in linguaggio macchina necessarie per un'agevole accesso ai registri dell'orologio contenuto in uno dei 6526. Esse sono state allocate a partire dall'indirizzo 9E00 ed occupano sino 9F12 ma per chi dovesse spostarle o modificarle e' riportato anche il listato in linguaggio assembly con i relativi commenti. Di qualche interesse possono essere le subroutine per la conversione dal codice BCD in ASCII e viceversa necessarie per adattare i dati del 6526 al Basic o video e viceversa. Il programma Basic, oltre a fornirne un esempio d'uso, contiene i relativi DATA per caricarle direttamente in memoria all'inizio di ogni seduta.

E' inoltre possibile attivare una particolare routine che

visualizzara' 50 volte al secondo, cioe' continuamente, l'ora esatta nell'angolo destro superiore dello schermo indipendentemente dai programmi in esecuzione. Questo puo' rivelarsi molto comodo quando sia necessario sapere in continuazione che ore sono durante il lavoro al computer. Naturalmente questa possibilita' puo' essere esclusa in ogni momento eseguendo la necessaria routine di disattivazione. Dopo l'attivazione non e' permesso ripeterla se non dopo la disattivazione, pena il blocco del funzionamento del computer dovuto alla perdita del vettore originale di interrupt, quindi un po' di attenzione non guasta. Per leggere l'ora durante l'esecuzione di un programma bisognera' definire una volta per tutte una variabile alfanumerica lunga otto caratteri necessaria per contenere tutte le cifre e richiamarla appena prima della routine di lettura. Tutto cio' vale anche in modo diretto da tastiera. Lo stesso andra' fatto per impostare l'ora di partenza e l'ora di allarme. Il programma Basic contiene gia' le necessarie subroutine di gestione quindi bastera' appenderle ai programmi applicativi e richiamarle quando necessario. In conclusione sarebbe veramente interessante sapere perche' il C64 non sfrutta questa possibilita' per contare il tempo invece di ricorrere alle interruzioni software con tutti i problemi visti. Forse tra i progettisti hardware e quelli software non corre buon sangue in casa Commodore.

REG	NOME	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
8	sec/10	0	0	0	0	d8	d4	d2	d1
9	sec	sh8	sh4	sh2	sh1	sl8	sl4	sl2	sl1
A	min	mh8	mh4	mh2	mh1	ml8	ml4	ml2	ml1
B	ore	PM	0	0	hh	hl8	hl4	hl2	hl1

h = parte alta l = parte bassa PM = indicazione AM/PM
bit del registro D = 1 ALLARME

```
1 rem     **************************
2 rem     *   programma di esempio  *
3 rem     * per l'uso dell'orologio *
4 rem     *     hardware del C64    *
5 rem     *  ing.l.squarza  03/85   *
6 rem     **************************
7 rem
10 gosub1000 :rem inizializza
20 gosub1400 :rem imposta l'ora di sveglia
30 gosub1500 :rem imposta l'ora attuale
40 gosub1750 :rem attiva la visualizzazione
99 end
100 rem******inserire qui il **********
110 rem******programma utente**********
120 rem
1000 rem
1001 rem***caricamento routine in ram**
1002 rem***e inizializzazione**********
1003 rem
1008 data  76 , 15 , 158 , 76 , 102 , 158 , 76 , 194 , 158 , 76
1010 data  217 , 158 , 76 , 242 , 158 , 8 , 160 , 1 , 177 , 71
1020 data  133 , 251 , 200 , 177 , 71 , 133 , 252 , 160 , 0 , 162
1030 data  3 , 189 , 8 , 221 , 16 , 12 , 201 , 146 , 240 , 8
1040 data  24 , 248 , 41 , 31 , 105 , 18 , 208 , 6 , 201 , 18
1050 data  240 , 240 , 41 , 31 , 32 , 83 , 158 , 202 , 48 , 6
1060 data  189 , 8 , 221 , 76 , 54 , 158 , 160 , 7 , 177 , 251
1070 data  72 , 136 , 177 , 251 , 200 , 145 , 251 , 136 , 104 , 145
1080 data  251 , 40 , 96 , 72 , 74 , 74
1085 data  74 , 74 , 9 , 48
1090 data  145 , 251 , 200 , 104 , 41 , 15 , 9 , 48 , 145 , 251
1100 data  200 , 96 , 8 , 160 , 1 , 177
1105 data  71 , 133 , 251 , 200
1110 data  177 , 71 , 133 , 252 , 162 , 3 , 160 , 0 , 32 , 177
1120 data  158 , 201 , 19 , 144 , 14 , 201 , 36 , 208 , 4 , 169
1130 data  146 , 208 , 6 , 248 , 56 , 233 , 18 , 9 , 128 , 157
1140 data  8 , 221 , 202 , 32 , 177 , 158 , 157 , 8 , 221 , 202
1150 data  32 , 177 , 158 , 157 , 8 , 221 , 202 , 32 , 177 , 158
1160 data  74 , 74 , 74 , 74 , 157 , 8 , 221 , 173 , 14 , 221
1170 data  9 , 128 , 141 , 14 , 221 , 40 , 96 , 177 , 251 , 10
1180 data  10 , 10 , 10 , 133 , 253 , 200 , 177 , 251 , 41 , 15
1190 data  5 , 253 , 200 , 96 , 173 , 15 , 221 , 9 , 128 , 141
1200 data  15 , 221 , 32 , 102 , 158 , 173 , 15 , 221 , 41 , 127
1210 data  141 , 15 , 221 , 173 , 13 , 221 , 96 , 120 , 173 , 20
1220 data  3 , 141 , 13 , 159 , 173 , 21 , 3 , 141 , 14 , 159
1230 data  169 , 1 , 141 , 20 , 3 , 169
1235 data  159 , 141 , 21 , 3
1240 data  88 , 96 , 120 , 173 , 13 , 159 , 141 , 20 , 3 , 173
1250 data  14 , 159 , 141 , 21 , 3 , 88
1255 data  96 , 169 , 32 , 133
1260 data  251 , 169 , 4 , 133 , 252 , 32 , 15 , 159 , 76 , 255
1270 data  255 , 8 , 76 , 27 , 158
1295 rem***indirizzo di cricamento 9E00*
1300 zz=9*16↑3+14*16↑2
1310 zh=int(zz/256):zl=zz-zh*256
1320 poke51,zl:poke52,zh:poke55,zl:poke56,zh
1325 rem***caricamento routine**********
1330 forx=zzto(zz+274)
1340 ready:pokex,y
1350 next
```

```
1360 return
1365 rem
1400 rem**** setta l'ora di allarme***
1410 print"S":input"qqqqÜÜÜallarme hhmmssdd";al$
1415 sb$=al$
1420
iflef t$(al$,2)="12"then:al$="24"+right$(al$,(len(al$)-2)):goto1450
1430 ifleft$(al$,2)="24"then:al$="12"+right$(al$,(len(al$)-2))
1450 al$=al$:sys40454
1455 al$=sb$
1490 return
1500 rem**** setta l'ora *************
1510 print"S":input"qqqqÜÜÜora     hhmmssdd";oo$
1550 oo$=oo$:sys40451
1590 return
1595 rem
1600 rem****lettura ora in tt$****
1610 tt$=tt$:sys40448
1620 return
1625 rem
1650 rem****test per allarme****
1660 rem se al=4 allora allarme
1670 al=(peek(56589)and4)
1680 return
1685 rem
1700 rem****attesa per allarme****
1710 rem esce solo dopo l'allarme
1720 wait56589,4:rem se il bit 2 =1 allora
1730 return :rem allarme
1735 rem
1750 rem****attiva la visualizzazione**
1760 rem****continua su video**********
1770 sys40457
1780 return
1785 rem
1800 rem****la disattiva*************
1810 sys40460
1820 return
```

FIG.16 Programma per il caricamento delle routine di gestione dell'orologio contenuto nel 6526 ed esempio d'uso.

```
LINE# LOC   CODE        LINE

00001  0000              ;      ***************************
00002  0000              ;      *  UTILIZZO DELL'OROLOGIO  *
00003  0000              ;      *    HARDWARE DEL C64      *
00004  0000              ;      ***************************
00005  0000              ;
00006  0000              ;DEFINIZIONE INDIRIZZI
00007  0000              ;
00008  0000              ORIG=$9F00                    ;INIZIO PROGRAMMA
00009  0000              TIMER=$DD08                   ;INDIRIZZO TIMER 6526
00010  0000              AREA=$FD                      ;LOCAZIONE USER
00011  0000              ARCOM=$FB                     ;VETTORE PER VARIABILE
00012  0000              INDIR=$47                     ;ULTIMA VARIABILE USATA
00013  0000              CREGA=$DD0E                   ;CRA DEL 6526
00014  0000              CREGB=$DD0F                   ;CRB DEL 6526
00015  0000              REGINT=$DD0D                  ;REGISTRO DELLE INTERRUZIONI
00016  0000              ;
00017  0000                     *=ORIG
00018  9F00   4C 09 9F          JMP RTIME              ;LEGGI ORA
00019  9F03   4C 60 9F          JMP STIME              ;SETTA ORA
00020  9F06   4C BC 9F          JMP SALAR              ;SETTA ALLARME
00021  9F09              ;
00022  9F09   08         RTIME  PHP                    ;SALVA STATUS REGISTER
00023  9F0A   A0 01             LDY #1                 ;PRENDE L'INDIRIZZO
00024  9F0C   B1 47             LDA (INDIR),Y          ;DELL'ULTIMA
00025  9F0E   85 FB             STA ARCOM              ;VARIABILE CHIAMATA
00026  9F10   C8                INY                    ;DA BASIC
00027  9F11   B1 47             LDA (INDIR),Y
00028  9F13   85 FC             STA ARCOM+1
00029  9F15   A0 00             LDY #0
00030  9F17   A2 03             LDX #3
00031  9F19   BD 08 DD          LDA TIMER,X            ;ORA AM/PM
00032  9F1C   10 0C             BPL POST               ;SE MSB=1 PM
00033  9F1E   C9 92      ANTE   CMP #$92               ;12 PM ALLORA ORE 12
00034  9F20   F0 08             BEQ POST
00035  9F22   18                CLC
00036  9F23   F8                SED                    ;MODO DECIMALE
00037  9F24   29 1F             AND #$1F               ;SOLO I 5 BIT BASSI
00038  9F26   69 12             ADC #$12               ;SOMMA 12 SE (PM)
00039  9F28   D0 06             BNE DAOR               ;VAI PURE
00040  9F2A   C9 12      POST   CMP #$12               ;SE 12 AM ALLORA 24
00041  9F2C   F0 F0             BEQ ANTE
00042  9F2E   29 1F             AND #$1F
00043  9F30   20 4D 9F   DAOR   JSR NUMA               ;CONVERTI IN ASCII
00044  9F33   CA                DEX                    ;PROSSIMO
00045  9F34   30 06             BMI FILOR              ;FINITO ?
00046  9F36   BD 08 DD          LDA TIMER,X            ;MINUTI,SECONDI,DECIMI
00047  9F39   4C 30 9F          JMP DAOR               ;RIPETI
00048  9F3C   A0 07      FILOR  LDY #7                 ;SISTEMA
00049  9F3E   B1 FB             LDA (ARCOM),Y          ;I DECIMI
00050  9F40   48                PHA                    ;AL POSTO
00051  9F41   88                DEY                    ;GIUSTO
00052  9F42   B1 FB             LDA (ARCOM),Y          ;INVERTENDOLI
00053  9F44   C8                INY                    ;CON I CENTESIMI
00054  9F45   91 FB             STA (ARCOM),Y
00055  9F47   88                DEY
```

```
LINE# LOC   CODE        LINE

00056  9F48  68                PLA
00057  9F49  91 FB             STA (ARCOM),Y
00058  9F4B  28                PLP              ;RIPRISTINA STATUS REGISTER
00059  9F4C  60                RTS
00060  9F4D               ;
00061  9F4D               ;CONVERSIONE DA BCD A ASCII
00062  9F4D  48           NUMA  PHA              ;SALVA IL NUMERO DA CONVERTIRE
00063  9F4E  4A                LSR A            ;PRENDE
00064  9F4F  4A                LSR A            ;LA
00065  9F50  4A                LSR A            ;PARTE
00066  9F51  4A                LSR A            ;ALTA
00067  9F52  09 30             ORA #$30         ;CONVERTE IN ASCII
00068  9F54  91 FB             STA (ARCOM),Y    ;METTE NELLA VARIABILE
00069  9F56  C8                INY              ;PROSSIMO CARATTERE
00070  9F57  68                PLA              ;RIPRENDE
00071  9F58  29 0F             AND #$0F         ;E CONVERTE
00072  9F5A  09 30             ORA #$30         ;LA PARTE
00073  9F5C  91 FB             STA (ARCOM),Y    ;BASSA DEL NUMERO
00074  9F5E  C8                INY
00075  9F5F  60                RTS
00076  9F60               ;
00077  9F60               ;IMPOSTA L'ORA NEL TIMER DEL 6526
00078  9F60               ;
00079  9F60  08           STIME PHP              ;SALVA STATUS
00080  9F61  A0 01             LDY #1           ;RECUPERA
00081  9F63  B1 47             LDA (INDIR),Y    ;L'INDIRIZZO
00082  9F65  85 FB             STA ARCOM        ;DELL'ULTIMA
00083  9F67  C8                INY              ;VARIABILE
00084  9F68  B1 47             LDA (INDIR),Y    ;CHIAMATA
00085  9F6A  85 FC             STA ARCOM+1      ;DA BASIC
00086  9F6C  A2 03             LDX #3
00087  9F6E  A0 00             LDY #0
00088  9F70  20 AB 9F          JSR ASCBC        ;CONVERTE DA ASCII A BCD
00089  9F73  C9 13        CONME CMP #$13         ;SE MINORE DI 13
00090  9F75  90 0E             BCC OKHH         ;ALLORA OK
00091  9F77  C9 24             CMP #$24         ;SE ORE 24
00092  9F79  D0 04             BNE MEN12
00093  9F7B  A9 92             LDA #$92         ;SETTA COME PM
00094  9F7D  D0 06             BNE OKHH         ;E VA PURE
00095  9F7F  F8           MEN12 SED              ;SE INVECE MAGGIORE
00096  9F80  38                SEC              ;ALLORA
00097  9F81  E9 12             SBC #$12         ;TOGLIE 12
00098  9F83  09 80        PM12  ORA #$80         ;E METTE IL FLAG DI PM
00099  9F85  9D 08 DD     OKHH  STA TIMER,X      ;METTE NEL TIMER
00100  9F88  CA                DEX              ;PROSSIMO
00101  9F89  20 AB 9F          JSR ASCBC        ;CONVERTE IN BCD
00102  9F8C  9D 08 DD          STA TIMER,X      ;E METTE NEL TIMER
00103  9F8F  CA                DEX              ;I MINUTI
00104  9F90  20 AB 9F          JSR ASCBC        ;CONVERTE I SECONDI
00105  9F93  9D 08 DD          STA TIMER,X      ;E LI METTE NEL TIMER
00106  9F96  CA                DEX              ;ADESSO
00107  9F97  20 AB 9F          JSR ASCBC        ;CONVERTE I DECIMI
00108  9F9A  4A                LSR A            ;LI
00109  9F9B  4A                LSR A            ;SPOSTA
00110  9F9C  4A                LSR A            ;AL POSTO
```

```
LINE# LOC    CODE          LINE


00111  9F9D  4A                     LSR A            ;GIUSTO
00112  9F9E  9D 08 DD               STA TIMER,X      ;E NEL TIMER
00113  9FA1  AD 0E DD               LDA CREGA        ;SETTA IL
00114  9FA4  09 80                  ORA #$80         ;FLAG DI 50 HZ
00115  9FA6  8D 0E DD               STA CREGA        ;NEL REGISTRO CRA
00116  9FA9  28                     PLP              ;RIPRISTINA LO STATUS
00117  9FAA  60                     RTS
00118  9FAB                  ;
00119  9FAB                  ;CONVERSIONE DA ASCII A BCD
00120  9FAB                  ;
00121  9FAB  B1 FB           ASCBC  LDA (ARCOM),Y    ;PRIMO CARATTERE
00122  9FAD  0A                     ASL A            ;LO SPOSTA
00123  9FAE  0A                     ASL A            ;SUI BIT ALTI
00124  9FAF  0A                     ASL A
00125  9FB0  0A                     ASL A
00126  9FB1  85 FD                  STA AREA         ;E LO SALVA
00127  9FB3  C8                     INY              ;PROSSIMO CARATTERE
00128  9FB4  B1 FB                  LDA (ARCOM),Y
00129  9FB6  29 0F                  AND #$0F         ;SOLO BIT BASSI
00130  9FB8  05 FD                  ORA AREA         ;LI UNISCE AI PRECEDENTI
00131  9FBA  C8                     INY              ;E OTTIENE IL NUMERO BCD
00132  9FBB  60                     RTS
00133  9FBC                  ;
00134  9FBC                  ;SET DELL'ORA DI ALLARME
00135  9FBC                  ;
00136  9FBC  AD 0F DD        SALAR  LDA CREGB        ;CRB DEL 6526
00137  9FBF  09 80                  ORA #$80         ;SE BIT 7 = 1 ALLORA
00138  9FC1  8D 0F DD               STA CREGB        ;SI SCRIVE IL REGISTRO
00139  9FC4  20 60 9F               JSR STIME        ;DI ALLARME E NON QUELLO
00140  9FC7  AD 0F DD               LDA CREGB        ;DELL'ORA
00141  9FCA  29 7F                  AND #$7F         ;RIMETTE TUTTO
00142  9FCC  8D 0F DD               STA CREGB        ;A POSTO
00143  9FCF  AD 0D DD               LDA REGINT       ;RESET DEL FLAG DI ALLARME
00144  9FD2  60                     RTS
00145  9FD3
00146  9FD3                         .END

ERRORS = 00000


SYMBOL TABLE

SYMBOL VALUE
 ANTE     9F1E     ARCOM    00FB     AREA     00FD     ASCBC    9FAB
 CONME    9F73     CREGA    DD0E     CREGB    DD0F     DAOR     9F30
 FILOR    9F3C     INDIR    0047     MEN12    9F7F     NUMA     9F4D
 OKHH     9F85     ORIG     9F00     PM12     9F83     POST     9F2A
 REGINT   DD0D     RTIME    9F09     SALAR    9FBC     STIME    9F60
 TIMER    DD08

END OF ASSEMBLY
U
```

I TRASDUTTORI

TEMPERATURA

TIPO:	TERMOSWITCH
CARATTERISTICHE ELETTRICHE	Chiusura o apertura di un contatto. Semplice controllo on/off.
COMMENTI:	Sono disponibili in molti tipi per diverse temperature, tipi di contatti e portate diverse

TIPO:	TERMOCOPPIA
CARATTERISTICHE ELETTRICHE	Bassa impedenza di uscita, tipica 10 OHM. Uscita in tensione dell' ordine di decimi di microvolt per grado centigrado.Uscita di millivolt a temperatura ambiente
COMMENTI:	La bassa tensione di uscita richiede sovente una amplificazione molto elevata e stabile.I vantaggi sono rappresentati dalle piccole dimensioni e dall' ampio range di temperatura. Richiede un riferimento a temperatura nota. La risposta e' non lineare

TIPO:	PLATINO E ALTRE RTD
CARATTERISTICHE ELETTRICHE	La resistenza varia con la temperatura Coefficiente di temp. positivo.Tipica impedenza (a 20 C) tra 20 Ohm e 2 KOhm Sensibilita' da 0.1%/C a 0.66%/C secondo il tipo di materiale
COMMENTI:	Ottima ripetibilita' e buona linearita'

	su ampi intervalli. Necessita normal- di connessione a ponte
TIPO:	TERMISTORI
CARATTERISTICHE ELETTRICHE	La resistenza varia con la temperatura. Coefficiente di temp negativo.Impedenze tipiche a 25 gradi disponibili tra 50 Ohm e 1 Mohm. Sensibilita' a 25 C circa 4% al grado.
COMMENTI:	Rispetto agli altri tipi presenta la piu' alta sensibilita'.Tipicamente non lineare (risposta esponenziale) ma sono disponibili accurate reti di linearizzazione
TIPO:	SENSORI A SEMICONDUTTORE
CARATTERISTICHE ELETTRICHE	Uscita disponibili sia in tensione o corrente che in resistenza.I tipi in tensione (diodi) richiedono una corrente di eccitazione.I tipi in corrente (AD590) richiedono una tensio-ne di eccitazione.I tipi a resistenza (BULK SILICON) possono essere alimentati in entrambi i modi.
COMMENTI:	Molti dispositivi non sono calibrati e richiedono una discreta amplificazione. L' AD590 e' calibrato,lineare e non richiede normalmente amplificazione

FORZA

TIPO:	STRAIN-GAGES METALLICI
CARATTERISTICHE ELETTRICHE	La resistenza varia in funzione della deformazione.Sono usati quasi sempre a ponte.L'impedenza tipica va da 120 Ohm a 350 Ohm.La variazione tipica e' l'1% nell'intero campo di misura.
COMMENTI:	La variazione di resistenza e' molto piccola rispetto al valore nominale. Richiede un ottimo amplificatore per bassi livelli di segnale
TIPO:	PONTI A STRAIN-GAGE,CELLE DI CARICO
CARATTERISTICHE ELETTRICHE	Uscita in tensione proporzionale al carico.Richiede una tensione od una corrente di eccitazione per il ponte normalmente compresa tra 5 e 15 Volt.
COMMENTI:	La bassa tensione di uscita richiede una buona e stabile amplificazione con immunita' di modo comune per sfruttarne l' alta pr cisione tipica. L' uscita e' lineare
TIPO:	STRAIN-GAGES A SEMICONDUTTORE
CARATTERISTICHE ELETTRICHE	Il tipo a ponte e' ottenuto con singoli strain-gage e l' uscita e' in tensione.Il ponte richiede eccitazione in tensione tra 5 e 15 Volt.
COMMENTI:	L'uscita e' di livello piu' alto rispetto ai tipi metallici ma anche superiore e' la non linearita' e la sensibilita'

	alla temperatura.
TIPO:	PIEZOELETTRICO
CARATTERISTICHE ELETTRICHE	Si puo′ considerare come un generatore di tensione in serie con una capacita′. Si ha in uscita una segnale solo durante le variazioni di carico.Frequenze di taglio superiori da 20 a 50 KHz.
COMMENTI:	Richiede una amplificatore ad alta impedenza di ingresso.Risponde solo alle variazioni di carico.

PRESSIONE

TIPO:	REOSTATO/POTENZIOMETRO
CARATTERISTICHE ELETTRICHE	L′uscita′ e′ resistenza o rapporto di rapporto di resistenze.Richiede una corrente od una tensione di eccitazione. Impedenza tipica da 500 Ohm a 5 KOhm.
COMMENTI:	L′uscita e′ ad alto livello e facile da elaborare ed e′ normalmente una ampia variazione di tensione o corrente.
TIPO:	STRAIN-GAGES
CARATTERISTICHE ELETTRICHE	vedere FORZA
COMMENTI:	" "
TIPO:	PIEZOELETTRICO
CARATTERISTICHE ELETTRICHE	vedere FORZA
COMMENTI:	" "

FLUSSO O PORTATA

TIPO: BASATI SU MISURE DI PRESSIONE

CARATTERISTICHE: vedere TRASDUTTORI DI PRESSIONE

COMMENTI: Questi tipidi sensori si basano sulla sulla misura della differenza di pressione tra due punti del fluido oppure oppure la caduta di pressione quando il fluido attraversa una strozzatura.Si usano dunque misuratori differenziali per evitare gli errori di modo comune. La risposta e' non lineare.

TIPO: USCITA IN FREQUENZA
PADDLE WHEELS
ROTANTI
VORTEX

CARATTERISTICHE ELETTRICHE: Comune a tutti i tipi e' l'uscita in frequenza a livello digitale.E' possibile il disaccoppiamento tra il trasduttore e l'elaboratore.Sono usati sia sensori magnetici che ottici con isolamento dal fluido.Le fotocellule presentano un rapporto on/off da 100 Ohm a 100 MOhm.I magnetici hanno una uscita un contatto od un transistor a collettore aperto.

COMMENTI: Alcuni tipi hanno un'uscita direttamente compatibile TTL,altri ppossono avere bisogno di amplificazione o traslazione di livello prima di essere utilizzabili

TIPO:	BASATI SU MISURE DI FORZA
CARATTERISTICHE ELETTRICHE	Sono normalmente usati STRAI-GAGES o POTENZIOMETRI, vedere PRESSIONE e FORZA
COMMENTI:	vedere PRESSIONE e FORZA
TIPO:	TERMICI
CARATTERISTICHE ELETTRICHE	Si basano su di un sensore attivo di temperatura per misurarne le variazioni causate dal flusso del fluido.
COMMENTI:	vedere TEMPERATURA

LIVELLO

TIPO:	A GALLEGGIANTE
CARATTERISTICHE ELETTRICHE	Utilizzano potenziometri o resistenze tra 100 Ohm e 2 KOhm.
COMMENTI:	Necessitano di eccitazione in tensione o in corrente per generare la tensione di uscita che si presenta di alto livello per le grandi variazioni di resistenza in gioco.
TIPO:	TERMICO
CARATTERISTICHE ELETTRICHE	Di tipo resistivo con impedenza tipica tra 500 Ohm e 2 KOhm.
COMMENTI:	Un sensore di temperatura auto-riscaldato e' usato per rilevare cambiamenti discreti nel livello delfluido.Una brusca variazione di temperatura viene rilevata quando il livello scende al punto di lasciare scoperto il sensore.

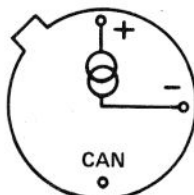
TIPO:	OTTICO
CARATTERISTICHE ELETTRICHE	Di tipo resistivo con tipica impedenza on/off tra 100 Ohm e 100 MOhm.
COMMENTI:	Il liquido viene usato per bloccare il passaggio della luce tra ricevitore e trasmettitore ottico.MIsura livelli discreti e non continui.
TIPO:	A PRESSIONE
CARATTERISTICHE ELETTRICHE	vedere PRESSIONE
COMMENTI:	L'informazione sul livello e' ricavata dalla differenza di pressione tra due sensori posti uno nella parte scoperta (superiore) e l'altro in quella occupata dal fluido (inferiore).
TIPO:	A CELLA DI CARICO
CARATTERISTICHE ELETTRICHE	vedere FORZA
COMMENTI:	L'informazione sul livello e' ottenuta dalla misura del peso dell'intero contenitore.

APPENDICE

Two-Terminal IC Temperature Transducer

AD590*

FEATURES
Linear Current Output: 1µA/°K
Wide Range: -55°C to +150°C
Probe Compatible Ceramic Sensor Package
Two-Terminal Device: Voltage In/Current Out
Laser Trimmed to ±0.5°C Calibration Accuracy (AD590M)
Excellent Linearity: ±0.3°C Over Full Range Range (AD590M)
Wide Power Supply Range: +4V to +30V
Sensor Isolation from Case
Low Cost

AD590 FUNCTIONAL BLOCK DIAGRAM

TO-52
BOTTOM VIEW

PRODUCT DESCRIPTION

The AD590 is a two-terminal integrated circuit temperature transducer which produces an output current proportional to absolute temperature. For supply voltages between +4V and +30V the device acts as a high impedance, constant current regulator passing 1µA/°K. Laser trimming of the chip's thin film resistors is used to calibrate the device to 298.2µA output at 298.2°K (+25°C).

The AD590 should be used in any temperature sensing application below +150°C in which conventional electrical temperature sensors are currently employed. The inherent low cost of a monolithic integrated circuit combined with the elimination of support circuitry makes the AD590 an attractive alternative for many temperature measurement situations. Linearization circuitry, precision voltage amplifiers, resistance measuring circuitry and cold junction compensation are not needed in applying the AD590.

In addition to temperature measurement, applications include temperature compensation or correction of discrete components, biasing proportional to absolute temperature, flow rate measurement, level detection of fluids and anemometry. The AD590 is available in chip form making it suitable for hybrid circuits and fast temperature measurements in protected environments.

The AD590 is particularly useful in remote sensing applications. The device is insensitive to voltage drops over long lines due to its high impedance current output. Any well-insulated twisted pair is sufficient for operation hundreds of feet from the receiving circuitry. The output characteristics also make the AD590 easy to multiplex: the current can be switched by a CMOS multiplexer or the supply voltage can be switched by a logic gate output.

*Covered by Patent No. 4,123,698

PRODUCT HIGHLIGHTS

1. The AD590 is a calibrated two terminal temperature sensor requiring only a dc voltage supply (+4V to +30V). Costly transmitters, filters, lead wire compensation and linearization circuits are all unnecessary in applying the device.
2. State-of-the-art laser trimming at the wafer level in conjunction with extensive final testing insures that AD590 units are easily interchangeable.
3. Superior interference rejection results from the output being a current rather than a voltage. In addition, power requirements are low (1.5mW's @ 5V @ +25°C). These features make the AD590 easy to apply as a remote sensor.
4. The high output impedance (>10MΩ) provides excellent rejection of supply voltage drift and ripple. For instance, changing the power supply from 5V to 10V results in only a 1µA maximum current change, or 1°C equivalent error.
5. The AD590 is electrically durable: it will withstand a forward voltage up to 44V and a reverse voltage of 20V. Hence, supply irregularities or pin reversal will not damage the device.
6. The device is hermetically sealed in both a ceramic sensor package and in to TO-52 package. MIL-STD-883 processing to level B is available and, for large unit volumes, special accuracy requirements over limited temperature ranges can be satisfied by selections at final test. The device is also available in chip form.

SPECIFICATIONS (typical @ +25°C and V_S = 5V unless otherwise noted)

MODEL	AD590I	AD590J	AD590K	AD590L	AD590M
ABSOLUTE MAXIMUM RATINGS					
Forward Voltage (E+ to E-)	+44V	*	*	*	*
Reverse Voltage (E+ to E-)	-20V	*	*	*	*
Breakdown Voltage (Case to E+ or E-)	±200V	*	*	*	*
Rated Performance Temperature Range[1]	-55°C to +150°C	*	*	*	*
Storage Temperature Range[1]	-65°C to +155°C	*	*	*	*
Lead Temperature (Soldering, 10 sec)	+300°C	*	*	*	*
POWER SUPPLY					
Operating Voltage Range	+4V to +30V	*	*	*	*
OUTPUT					
Nominal Current Output @ +25°C (298.2°K)	298.2μA	*	*	*	*
Nominal Temperature Coefficient	1μA/°K	*	*	*	*
Calibration Error @ +25°C	±10.0°C max	±5.0°C max	±2.5°C max	±1.0°C max	±0.5°C max
Absolute Error[2] (over rated performance temperature range)					
Without External Calibration Adjustment	±20.0°C max	±10.0°C max	±5.5°C max	±3.0°C max	±1.7°C max
With +25°C Calibration Error Set to Zero	±5.8°C max	±3.0°C max	±2.0°C max	±1.6°C max	±1.0°C max
Nonlinearity	±3.0°C max	±1.5°C max	±0.8°C max	±0.4°C max	±0.3°C max
Repeatability[3]	±0.1°C max	*	*	*	*
Long Term Drift[4]	±0.1°C/month max	*	*	*	*
Current Noise	40pA/$\sqrt{Hz}$	*	*	*	*
Power Supply Rejection					
$+4V \leq V_S \leq +5V$	0.5μA/V	*	*	*	*
$+5V \leq V_S \leq +15V$	0.2μA/V	*	*	*	*
$+15V \leq V_S \leq +30V$	0.1μA/V	*	*	*	*
Case Isolation to Either Lead	$10^{10}\Omega$	*	*	*	*
Effective Shunt Capacitance	100pF	*	*	*	*
Electrical Turn-On Time[5]	20μs	*	*	*	*
Reverse Bias Leakage Current[6] (Reverse Voltage = 10V)	10pA	*	*	*	*
PACKAGE OPTION[7]					
"H" Package: TO-52	AD590IH	AD590JH	AD590KH	AD590LH	AD590MH
"F" Package: Flat Pack (F2A)	AD590IF	AD590JF	AD590KF	AD590LF	AD590MF

*Specifications same as AD590I

[1] The AD590 has been used at -100°C and +200°C for short periods of measurement with no physical damage to the device. However, the absolute errors specified apply to only the rated performance temperature range.

[2] See section on temperature sensor specifications for explanation of error components. Note that ±1°C error is the equivalent of ±1μA error.

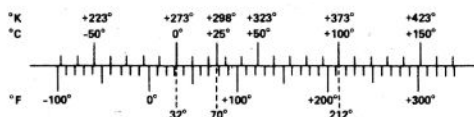
[3] Maximum deviation between +25°C readings after temperature cycling between -55°C and +150°C; guaranteed not tested.

[4] Conditions: constant +5V, constant +125°C; guaranteed, not tested.

[5] Does not include self heating effects.

[6] Leakage current doubles every 10°C.

[7] See Section 20 for package outline information.

Specifications subject to change without notice.

TEMPERATURE SCALE CONVERSION EQUATIONS

$$°C = \frac{5}{9}(°F - 32) \qquad °K = °C + 273.15$$

$$°F = \frac{9}{5}°C + 32 \qquad °R = °F + 459.7$$

CIRCUIT DESCRIPTION[1]

The AD590 uses a fundamental property of the silicon transistors from which it is made to realize its temperature proportional characteristic: if two identical transistors are operated at a constant ratio of collector current densities, r, then the difference in their base-emitter voltages will be (kT/q)(In r). Since both k, Boltzman's constant and q, the charge of an electron, are constant, the resulting voltage is directly proportional to absolute temperature (PTAT).

In the AD590, this PTAT voltage is converted to a PTAT current by low temperature coefficient thin film resistors. The total current of the device is then forced to be a multiple of this PTAT current. Referring to Figure 1, the schematic diagram of the AD590, Q8 and Q11 are the transistors that produce the PTAT voltage. R5 and R6 convert the voltage to current. Q10, whose collector current tracks the collector currents in Q9 and Q11, supplies all the bias and substrate leakage current for the rest of the circuit, forcing the total current to be PTAT. R5 and R6 are laser trimmed on the wafer to calibrate the device at +25°C.

Figure 1. Schematic Diagram

Figure 2 shows the typical V−I characteristic of the circuit at +25°C and the temperature extremes.

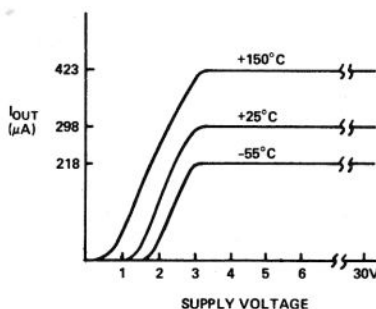



Figure 2. V−I Plot

[1] For a more detailed circuit description see M.P. Timko, "A Two-Terminal IC Temperature Transducer," IEEE J. Solid State Circuits, Vol. SC-11, p. 784-788, Dec. 1976.

MC14099B
MC14599B

8-BIT ADDRESSABLE LATCHES

The MC14099B and MC14599B are 8-bit addressable latches. Data is entered in serial form when the appropriate latch is addressed (via address pins A0, A1, A2) and write disable is in the low state. Chip enable must be high for writing into MC14599B. For the MC14599B the data pin is a bidirectional data port and for the MC14099B the input is a unidirectional write only port. The Write/$\overline{\text{Read}}$ line controls this port in the MC14599B.

The data is presented in parallel at the output of the eight latches independently of the state of Write Disable, Write/$\overline{\text{Read}}$ or Chip Enable.

A Master Reset capability is available on both parts.

- Serial Data Input
- Parallel Output
- Low Input Capacitance – 5.0 pF typical
- Master Reset
- Noise Immunity – 45% of V_{DD} typical
- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- Capable of Driving Two Low-Power TTL Loads, One Low-Power Schottky TTL Load or Two HTL Loads over the Rated Temperature Range
- MC14099B pin for pin compatible with CD4099B
- Pin for pin compatible with Fairchild 4724.

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	-0.5 to +18	Vdc
Input Voltage, All Inputs	V_{in}	-0.5 to V_{DD} + 0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range – AL Device CL/CP Device	T_A	-55 to +125 -40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

CMOS MSI

(LOW-POWER COMPLEMENTARY MOS)

8-BIT ADDRESSABLE LATCH

MC14599B WITH BIDIRECTIONAL PORT

L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

L SUFFIX
CERAMIC PACKAGE
CASE 680

P SUFFIX
PLASTIC PACKAGE
CASE 707

ORDERING INFORMATION

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	V_{DD} Vdc	T_{low}* Min	T_{low}* Max	25°C Min	25°C Typ	25°C Max	T_{high}* Min	T_{high}* Max	Unit
Output Voltage "0" Level $V_{in} = V_{DD}$ or 0	V_{OL}	5.0	–	0.05	–	0	0.05	–	0.05	Vdc
		10	–	0.05	–	0	0.05	–	0.05	
		15	–	0.05	–	0	0.05	–	0.05	
"1" Level $V_{in} = 0$ or V_{DD}	V_{OH}	5.0	4.95	–	4.95	5.0	–	4.95	–	Vdc
		10	9.95	–	9.95	10	–	9.95	–	
		15	14.95	–	14.95	15	–	14.95	–	
Input Voltage# "0" Level (V_O = 4.5 or 0.5 Vdc)	V_{IL}	5.0	–	1.5	–	2.25	1.5	–	1.5	Vdc
(V_O = 9.0 or 1.0 Vdc)		10	–	3.0	–	4.50	3.0	–	3.0	
(V_O = 13.5 or 1.5 Vdc)		15	–	4.0	–	6.75	4.0	–	4.0	
"1" Level (V_O = 0.5 or 4.5 Vdc)	V_{IH}	5.0	3.5	–	3.5	2.75	–	3.5	–	Vdc
(V_O = 1.0 or 9.0 Vdc)		10	7.0	–	7.0	5.50	–	7.0	–	
(V_O = 1.5 or 13.5 Vdc)		15	11.0	–	11.0	8.25	–	11.0	–	
Output Drive Current (AL Device) (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	-3.0	–	-2.4	-4.2	–	-1.7	–	mAdc
(V_{OH} = 4.6 Vdc)		5.0	-0.64	–	-0.51	-0.88	–	-0.36	–	
(V_{OH} = 9.5 Vdc)		10	-1.6	–	-1.3	-2.25	–	-0.9	–	
(V_{OH} = 13.5 Vdc)		15	-4.2	–	-3.4	-8.8	–	-2.4	–	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	–	0.51	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	–	1.3	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)		15	4.2	–	3.4	8.8	–	2.4	–	
Output Drive Current (CL/CP Device) (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	-2.5	–	-2.1	-4.2	–	-1.7	–	mAdc
(V_{OH} = 4.6 Vdc)		5.0	-0.52	–	-0.44	-0.88	–	-0.36	–	
(V_{OH} = 9.5 Vdc)		10	-1.3	–	-1.1	-2.25	–	-0.9	–	
(V_{OH} = 13.5 Vdc)		15	-3.6	–	-3.0	-8.8	–	-2.4	–	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.52	–	0.44	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)		10	1.3	–	1.1	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)		15	3.6	–	3.0	8.8	–	2.4	–	
Input Current (AL Device)	I_{in}	15	–	±0.1	–	±0.00001	±0.1	–	±1.0	µAdc
Input Current (CL/CP Device)	I_{in}	15	–	±0.3	–	±0.00001	±0.3	–	±1.0	µAdc
Input Capacitance (V_{in} = 0)	C_{in}	–	–	–	–	5.0	7.5	–	–	pF
Input Capacitance MC14599B – Data (pin 3) (V_{in} = 0)	C_{in}	–	–	–	–	15.0	22.5	–	–	pF
Quiescent Current (AL Device) (Per Package)	I_{DD}	5.0	–	5.0	–	0.005	5.0	–	150	µAdc
		10	–	10	–	0.010	10	–	300	
		15	–	20	–	0.015	20	–	600	
Quiescent Current (CL/CP Device) (Per Package)	I_{DD}	5.0	–	20	–	0.005	20	–	150	µAdc
		10	–	40	–	0.010	40	–	300	
		15	–	80	–	0.015	80	–	600	
Total Supply Current**† Dynamic plus Quiescent, Per Package) (C_L = 50 pF on all outputs, all buffers switching)	I_T	5.0	I_T = (1.5 µA/kHz) f + I_{DD}							µAdc
		10	I_T = (3.0 µA/kHz) f + I_{DD}							
		15	I_T = (4.5 µA/kHz) f + I_{DD}							

*T_{low} = -55°C for AL Device, -40°C for CL/CP Device.
T_{high} = +125°C for AL Device, +85°C for CL/CP Device.

#Noise immunity specified for worst-case input combination.
Noise Margin for both "1" and "0" level =
1.0 Vdc min @ V_{DD} = 5.0 Vdc
2.0 Vdc min @ V_{DD} = 10 Vdc
2.5 Vdc min @ V_{DD} = 15 Vdc

†To calculate total supply current at loads other than 50 pF:

$$I_T(C_L) = I_T(50\text{ pF}) + 4 \times 10^{-3}(C_L - 50)\, V_{DD} f$$

where: I_T is in µA (per package), C_L in pF, V_{DD} in Vdc, and f in kHz is input frequency.

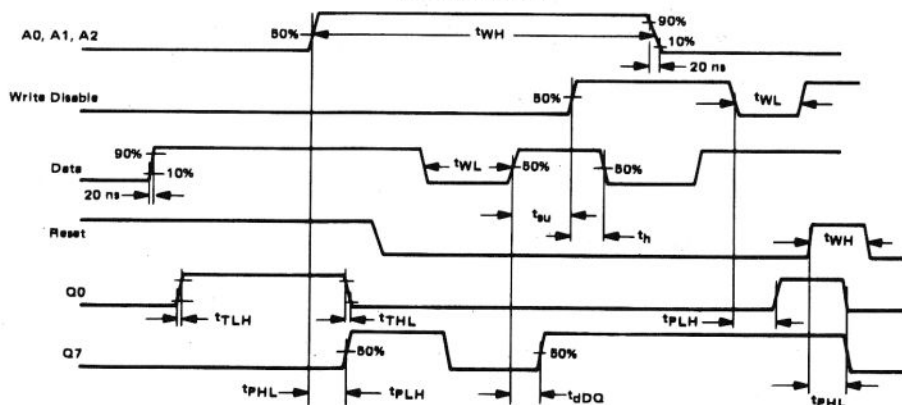
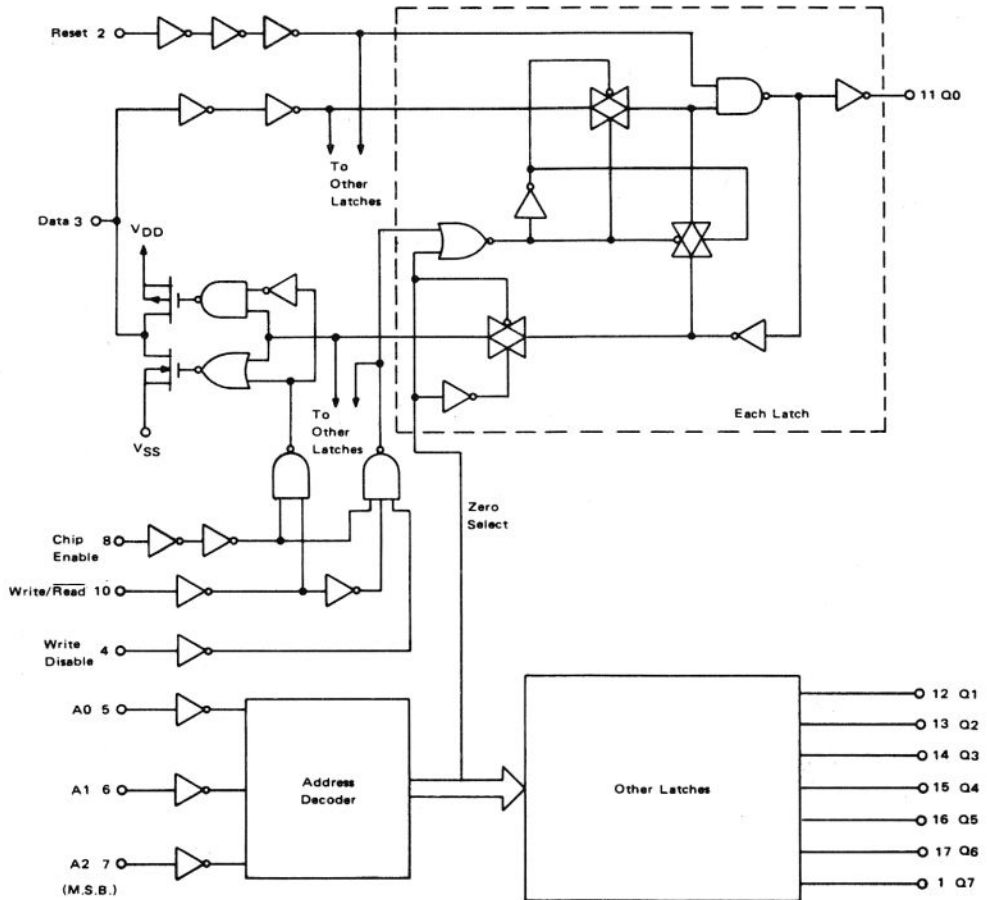
**The formulas given are for the typical characteristics only at 25°C.

SWITCHING CHARACTERISTICS (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD} Vdc	Min	Typ	Max	Unit
Output Rise and Fall Time	t_{TLH}, t_{THL}					ns
t_{TLH}, t_{THL} = (1.35 ns/pF) C_L +32 ns		5.0	–	100	200	
t_{TLH}, t_{THL} = (0.6 ns/pF) C_L +20 ns		10	–	50	100	
t_{TLH}, t_{THL} = (0.4 ns/pF) C_L +20 ns		15	–	40	80	
Propagation Delay Time	t_{PHL}, t_{PLH}					ns
Data to Output		5.0	–	200	400	
		10	–	75	150	
		15	–	50	100	
Write Disable to Output		5.0	–	200	400	ns
		10	–	80	160	
		15	–	60	120	
Reset to Output		5.0	–	175	350	ns
		10	–	80	160	
		15	–	65	130	
Address, CE to Output		5.0	–	225	450	ns
		10	–	100	200	
		15	–	75	150	
Propagation Delay Time, MC14599B only	t_{PHL}, t_{PLH}					ns
Chip Enable, Write/$\overline{\text{Read}}$ to Data		5.0	–	200	400	
		10	–	80	160	
		15	–	65	130	
Address to Data		5.0	–	200	400	ns
		10	–	90	180	
		15	–	75	150	
Minimum Pulse Widths	t_{WH}, t_{WL}					ns
Data		5.0	200	100	–	
		10	100	50	–	
		15	80	40	–	
Address		5.0	400	200	–	ns
		10	200	100	–	
		15	125	65	–	
Reset		5.0	150	75	–	ns
		10	75	40	–	
		15	50	25	–	
Write Disable		5.0	320	160	–	ns
		10	160	80	–	
		15	120	60	–	
Set Up Time	t_{su}					ns
Data		5.0	100	50	–	
		10	50	25	–	
		15	35	20	–	
Hold Time	t_h					ns
Data		5.0	150	75	–	
		10	75	40	–	
		15	50	25	–	

TRUTH TABLE

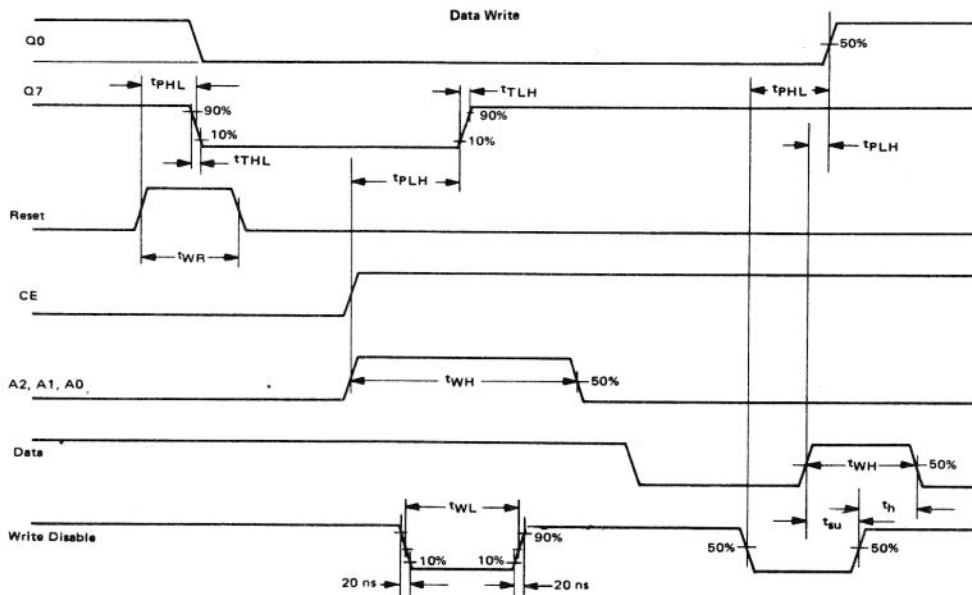
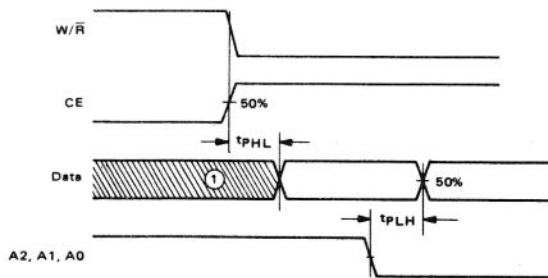
Write Disable	Reset	Addressed Latch	Unaddressed Latch
0	0	Data	Q_n*
0	1	Data	Reset†
1	0	Q_n*	Q_n*
1	1	Reset	Reset

* Q_n is previous state of latch.
† Reset to zero state.

TIMING DIAGRAM

TRUTH TABLE

Chip Enable	Write/$\overline{\text{Read}}$	Write Disable	Reset	Addressed Latch	Other Latches	Data Pin
0	X	X	0	•	•	Z
1	1	0	0	Data	•	Input
1	1	1	0	•	•	Z
1	0	X	0	•	•	Q_n
X	X	X	1	0	0	Z/0

X = Don't care.
• = No change in state of latch.
Z = High impedance.
Q_n = State of addressed latch.

MC14599B
TIMING DIAGRAMS

Data Write

Data Read

NOTE: 1. Invalid Data Output
2. Reset in LOW State

MC14433

3½ DIGIT A/D CONVERTER

The MC14433 is a high performance, low power, 3½ digit A/D converter combining both linear CMOS and digital CMOS circuits on a single monolithic IC. The MC14433 is designed to minimize use of external components. With two external resistors and two external capacitors, the system forms a dual slope A/D converter with automatic zero correction and automatic polarity.

The MC14433 is ratiometric and may be used over a full-scale range from 1.999 volts to 199.9 millivolts. Systems using the MC14433 may operate over a wide range of power supply voltages for ease of use with batteries, or with standard 5 volt supplies. The output drive conforms with standard B-Series CMOS specifications and can drive a low-power Schottky TTL load.

The high impedance MOS inputs allow applications in current and resistance meters as well as voltmeters. In addition to DVM/DPM applications, the MC14433 finds use in digital thermometers, digital scales, remote A/D, A/D control systems, and in MPU systems.

- Accuracy: ±0.05% of Reading ±1 Count
- Two Voltage Ranges: 1.999 V and 199.9 mV
- Up to 25 Conversions/s
- Z_{in} > 1000 M ohm
- Auto-Polarity and Auto-Zero
- Single Positive Voltage Reference
- Standard B-Series CMOS Outputs—Drives One Low Power Schottky Load
- Uses On-Chip System Clock, or External Clock
- Low Power Consumption: 8.0 mW typical @ ±5.0 V
- Wide Supply Range: e.g., ±4.5 V to ±8.0 V
- Overrange and Underrange Signals Available
- Operates in Auto Ranging Circuits
- Operates with LED and LCD Displays
- Low External Component Count

CMOS LSI

(LOW-POWER COMPLEMENTARY MOS)

3½ DIGIT A/D CONVERTER

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD} to V_{EE}	-0.5 to +18	Vdc
Voltage, any pin, referenced to V_{EE}	V	-0.5 to V_{DD} +0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range	T_A	-40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{EE} \leqslant (V_{in}$ or $V_{out}) \leqslant V_{DD}$.

RECOMMENDED OPERATING CONDITIONS (V_{SS} = 0 or V_{EE})

Parameter	Symbol	Value	Unit
DC Supply Voltage – V_{DD} to Analog Ground	V_{DD}	+5.0 to +8.0	Vdc
V_{EE} to Analog Ground	V_{EE}	-2.8 to -8.0	
Clock Frequency	f_{Clk}	32 to 400	kHz
Zero Offset Correction Capacitor	C_o	0.1 ± 20%	µF

ELECTRICAL CHARACTERISTICS

(C_I = 0.1 µF mylar, R_I = 470 kΩ @ V_{ref} = 2.000 V, R_I = 27 kΩ @ V_{ref} = 200.0 mV, C_o = 0.1 µF, R_C = 300 kΩ; all voltages referenced to Analog Ground, pin 1.)

Characteristic	Symbol	V_{DD} Vdc	V_{EE} Vdc	-40°C Min	-40°C Max	25°C Min	25°C Typ	25°C Max	85°C Min	85°C Max	Unit
Linearity-Output Reading (Note 1)	–										%rdg
(V_{ref} = 2.000 V)		5.0	-5.0	–	–	-0.05 -Count	±0.05	+0.05 +Count	–	–	
(V_{ref} = 200.0 mV)		5.0	-5.0	–	–	–	±0.05	–	–	–	
Stability-Output Reading (Note 2)	–										LSD
(V_X = 1.990 V, V_{ref} = 2.000 V)		5.0	-5.0	–	–	–	–	2	–	–	
(V_X = 199.0 mV, V_{ref} = 200.0 mV)		5.0	-5.0	–	–	–	–	3	–	–	
Zero-Output Reading (V_X = 0 V, V_{ref} = 2.000 V)	–	5.0	-5.0	–	–	–	0	0	–	–	LSD
Bias Current – Analog Input	–	5.0	-5.0	–	–	–	±20	±100	–	–	pAdc
Reference Input		5.0	-5.0	–	–	–	±20	±100	–	–	
Analog Ground		5.0	-5.0	–	–	–	±20	±500	–	–	
Common Mode Rejection (V_X = 1.4 V, V_{ref} = 2.000 V, f_{oc} = 32 kHz)	–	5.0	-5.0	–	–	–	65	–	–	–	dB
Output Voltage – Pins 14 to 23											Vdc
(V_{SS} = 0 V) "0" Level	V_{OL}	5.0	-5.0	–	0.05	–	0	0.05	–	0.05	
"1" Level	V_{OH}	5.0	-5.0	4.95	–	4.95	5.0	–	4.95	–	
(V_{SS} = -5.0 V) "0" Level	V_{OL}	5.0	-5.0	–	4.95	–	-5.0	-4.95	–	-4.95	
"1" Level	V_{OH}	5.0	-5.0	4.95	–	4.95	5.0	–	4.95	–	
Output Current – Pins 14 to 23											mAdc
(V_{SS} = 0 V)											
(V_{OH} = 4.6 V) Source	I_{OH}	5.0	-5.0	-0.25	–	-0.2	-0.36	–	-0.14	–	
(V_{OL} = 0.4 V) Sink	I_{OL}	5.0	-5.0	0.64	–	0.51	0.88	–	0.36	–	
(V_{SS} = -5.0 V)											
(V_{OH} = 4.5 V) Source	I_{OH}	5.0	-5.0	-0.62	–	-0.5	-0.9	–	-0.35	–	
(V_{OL} = -4.5 V) Sink	I_{OL}	5.0	-5.0	1.6	–	1.3	2.25	–	0.9	–	
Clock Frequency (R_C = 300 kΩ)	f_{Clk}	5.0	-5.0	–	–	–	66	–	–	–	kHz
Input Current – DU	I_{DU}	5.0	-5.0	–	±0.3	–	±0.00001	±0.3	–	±1.0	µAdc
Quiescent Current (V_{DD} to V_{EE}, I_{SS} = 0)	I_Q	5.0	-5.0	–	3.7	–	0.9	2.0	–	1.6	mAdc
		8.0	-8.0	–	7.4	–	1.8	4.0	–	3.2	
DC Supply Rejection (V_{DD} to V_{EE}, I_{SS} = 0, V_{ref} = 2.000V)	–	5.0	-5.0	–	–	–	0.5	–	–	–	mV/V

Note 1: Accuracy – The accuracy of the meter at full scale is the accuracy of the setting of the reference voltage. Zero is recalculated during each conversion cycle. The meaningful specification is linearity. In other words, the deviation from correct reading for all inputs other than positive full scale and zero is defined as the linearity specification.

Note 2: 3 LSD stability for 200 mV scale is defined as the range that the LSD will occupy 95% of the time.

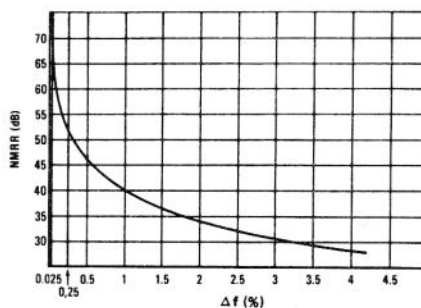
TYPICAL CHARACTERISTICS

TYPICAL CHARACTERISTICS

FIGURE (Table) 7 – FOR BEST 50 Hz
NMRR (Normal Mode Rejection Ratio)
AT DIFFERENT CLOCK FREQUENCIES

Clock frequency (KHz)	Integration time (msec)	Conversion rate (~ Hz)
200	20	12.5
100	40	6.25
66.6	60	4.16
50	80	3.12
40	100	2.5

The NMRR depends on the difference Δf (°/o) between a multiple of the power line frequency and the clock frequency.

FIGURE 8 – NMRR (dB) vs Δf (%)

DEVICE OPERATION

ANALOG GROUND (V_{AG}, Pin 1)

Analog ground at this pin is the input reference level for the unknown input voltage (V_X) and reference voltage (V_{ref}). This pin is a high impedance input.

REFERENCE VOLTAGE (V_{ref}, Pin 2)
UNKNOWN INPUT VOLTAGE (V_X, Pin 3)

This A/D system performs a ratiometric A/D conversion; that is, the unknown input voltage, V_X, is measured as a ratio of the reference voltage, V_{ref}. The full scale voltage is equal to that voltage applied to V_{ref}. Therefore, a full scale voltage of 1.999 V requires a reference voltage of 2.000 V while full scale voltage of 199.9 mV requires a reference voltage of 200 mV. Both V_X and V_{ref} are high impedance inputs. In addition to being a reference input, pin 2 functions as a reset for the A/D converter. When pin 2 is switched to V_{EE} for at least 5 clock cycles, the system is reset to the beginning of a conversion cycle.

EXTERNAL COMPONENTS (R_I, R_I/C_I, C_I; Pins 4, 5, 6)

These pins are for external components for the integration used in the dual ramp A/D conversion. A typical value for the capacitor is 0.1 μF (mylar) while the resistor should be 470 kΩ for 2.0 V full scale operation and 27 kΩ for 200 mV full scale operation. These values are for a 66 kHz clock frequency which will produce a conversion time of approximately 250 ms. The equations governing the calculation for the values for integrator components are as follows:

$$R_I = \frac{V_X(max)}{C_I} \times \frac{T}{\Delta V}$$

$$\Delta V = V_{DD} - V_X(max) - 0.5$$

$$T = 4000 \times \frac{1}{f_{Clk}}$$

where:

R_I is in kΩ
V_{DD} is the voltage at pin 24 referenced to V_{AG}
V_X is the voltage at pin 3 referenced to V_{AG}
f_{Clk} is the clock frequency at pin 10 in kHz

Example:
$C_I = 0.1\ \mu F$
$V_{DD} = 5.0$ volts
$f_{Clk} = 66$ kHz
For $V_X(max) = 2.0$ volts
$R_I = 480$ kΩ (use 470 k$\Omega \pm 5\%$)
For $V_X(max) = 200$ mV
$R_I = 28$ kΩ (use 27 k$\Omega \pm 5\%$)

Note that for worst case conditions, the minimum allowable value for R_I is a function of C_I min, V_{DD} min, and f_{Clk} max. The worst-case condition does not allow $V + V_X$ to exceed V_{DD}. The 0.5 V factor in the above equation for ΔV is for safety margin.

OFFSET CAPACITOR (CO1, CO2; Pins 7, 8)

These pins are used for connecting the offset correction capacitor. The recommended value is 0.1 μF.

DISPLAY UPDATE INPUT (DU, Pin 9)

If a positive edge is received on this input prior to the ramp-down cycle, new data will be strobed into the output latches during that conversion cycle. When this pin is wired directly to the EOC output (pin 14), every conversion will be displayed. When this pin is driven from an external source, the voltage should be referenced to V_{SS}.

CLOCK (Clk I, Clk O, Pins 10, 11)

The MC14433 device contains its own oscillator system clock. A single resistor connected between pins 10 and 11 sets the clock frequency. If increased stability is desired, these pins will support a crystal or LC circuit. The clock input, pin 10, may also be driven from an external clock source which need have only standard CMOS output drive. For external clock inputs this pin is referenced to V_{EE}. A 300 kΩ resistor results in clock frequency of about 66 kHz. (See the typical characteristic curves.) For alternate circuits see Figure 7.

NEGATIVE POWER SUPPLY (V_{EE}, Pin 12)

This is the connection for the most negative power supply voltage. The typical current is 0.8 mA. Note the current for the output drive circuit is not returned through this pin, but through pin 13.

NEGATIVE POWER SUPPLY FOR OUTPUT CIRCUITRY (V_{SS}, Pin 13)

This is the low voltage level for the output pins of the MC14433 (BCD, Digit Selects, EOC, $\overline{OR}$). When this pin is connected to analog ground, the output voltage is from analog ground to V_{DD}. When connected to V_{EE}, the output swing is from V_{EE} to V_{DD}. The allowable operating range for V_{SS} is between $V_{DD} - 3.0$ volts and V_{EE}.

END OF CONVERSION (EOC, Pin 14)

The EOC output produces a pulse at the end of each conversion cycle. This pulse width is equivalent to one half the period of the system clock (pin 11).

OVERRANGE ($\overline{OR}$, Pin 15)

The $\overline{OR}$ pin is low when V_X exceeds V_{ref}. Normally it is high.

DIGIT SELECT (DS4, DS3, DS2, DS1; Pins 16, 17, 18, 19)

The digit select output is high when the respective

digit is selected. The most significant digit (½ digit) turns on immediately after an EOC pulse followed by the remaining digits, sequencing from MSD to LSD. An inter-digit blanking time of two clock periods is included to ensure that the BCD data has settled. The multiplex rate is equal to the clock frequency divided by 80. Thus, with a system clock rate of 66 kHz, the multiplex rate would be 0.8 kHz. Relative timing among digital select ouptut and EOC signals is shown in the Digit Select Timing Diagram, Figure 8.

BCD DATA OUTPUTS (Q0, Q1, Q2, Q3, Pins 20, 21, 22, 23)

Multiplexed BCD outputs contain 3 full digits of information during DS2, 3, 4, while during DS1, the ½ digit, overrange, underrange and polarity are available. The adjacent truth table shows the formats of the information during DS1.

POSITIVE POWER SUPPLY (V_{DD}, Pin 24)

The most positive supply voltage pin.

TRUTH TABLE

Coded Condition of MSD	Q3	Q2	Q1	Q0	BCD to 7 Segment Decoding
+0	1	1	1	0	Blank
−0	1	0	1	0	Blank
+0 UR	1	1	1	1	Blank
−0 UR	1	0	1	1	Blank
+1	0	1	0	0	4 → 1 Hook up
−1	0	0	0	0	0 → 1 only seg b
+1 OR	0	1	1	1	7 → 1 and c to
−1 OR	0	0	1	1	3 → 1 MSD

Notes for Truth Table
Q3 – ½ digit, low for "1", high for "0"
Q2 – Polarity: "1" = positive, "0" = negative
Q0 – Out of range condition exists if Q0 = 1. When used in conjunction with Q3 the type of out of range condition is indicated, i.e., Q3 = 0 → OR or Q3 = 1 → UR.

When only segment b and c of the decoder are connected to the ½ digit of the display, 4, 0, 7 and 3 appear as 1.

The overrange indication (Q3 = 0 and Q0 = 1) occurs when the count is greater than 1999, e.g., 1.999 V for a reference of 2.000 V. The underrange indication, useful for autoranging circuits, occurs when the count is less than 180, e.g., 0.180 V for a reference of 2.000 V.

Caution: If the most significant digit is connected to a display other than a "1" only; such as a full digit display, segments other than b and c must be disconnected. The BCD to seven segment decoder must blank on BCD inputs 1010 to 1111.

FIGURE 7 – ALTERNATE OSCILLATOR CIRCUITS

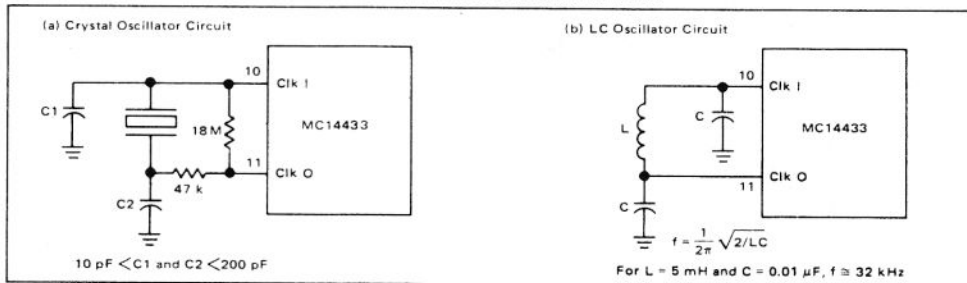



FIGURE 8 – DIGIT SELECT TIMING DIAGRAM

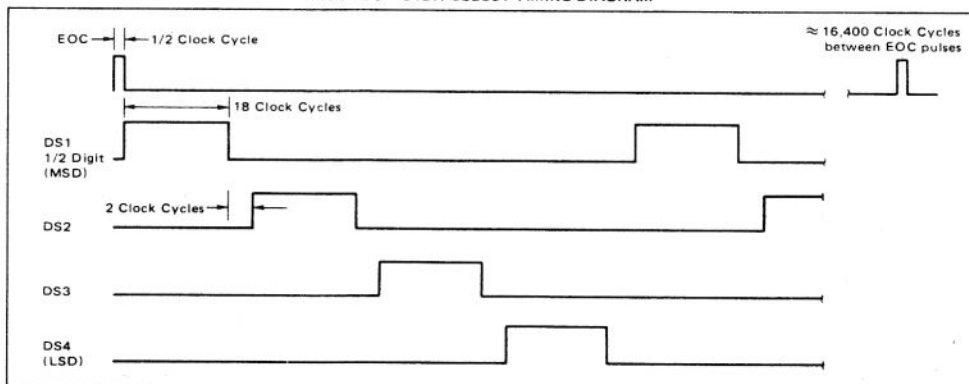



FIGURE 9 – INTEGRATOR WAVEFORMS AT PIN 6

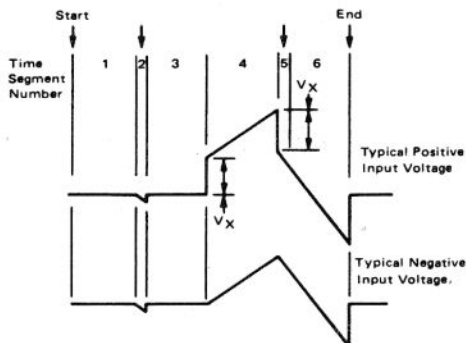
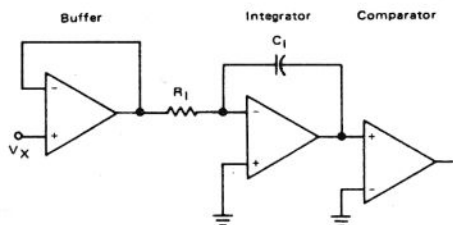



FIGURE 10 – EQUIVALENT CIRCUIT DIAGRAMS OF THE ANALOG SECTION DURING SEGMENT 4 OF THE TIMING CYCLE

CIRCUIT OPERATION

The MC14433 CMOS integrated circuit, together with a minimum number of external components, forms a modified dual ramp A/D converter. The device contains the customary CMOS digital logic providing counters, latches, and multiplexing circuitry as well as the CMOS analog circuitry providing operational amplifiers and comparators required to implement a complete single chip A/D. Autozero, high input impedances, and autopolarity are features of this system. Using CMOS technology, an A/D with a wide range of power supply voltage and low power consumption is now available with the MC14433.

During each conversion, the offset voltages of the internal amplifiers and comparators are compensated for by the system's autozero operation. Also each conversion 'ratiometrically' measures the unknown input voltage. In other words, the output reading is the ratio of the unknown voltage to the reference voltage with a ratio of 1 equal to the maximum count 1999. The entire conversion cycle requires slightly more than 16000 clock periods and may be divided into six different segments. The waveforms showing the conversion cycle with a positive input and a negative input are shown in Figure 9. The six segments of these waveforms are described below.

Segment 1 – The offset capacitor (C_O), which compensates for the input offset voltages of the buffer and integrator amplifiers, is charged during this period. Also, the integrator capacitor is shorted. This segment requires 4000 clock periods.

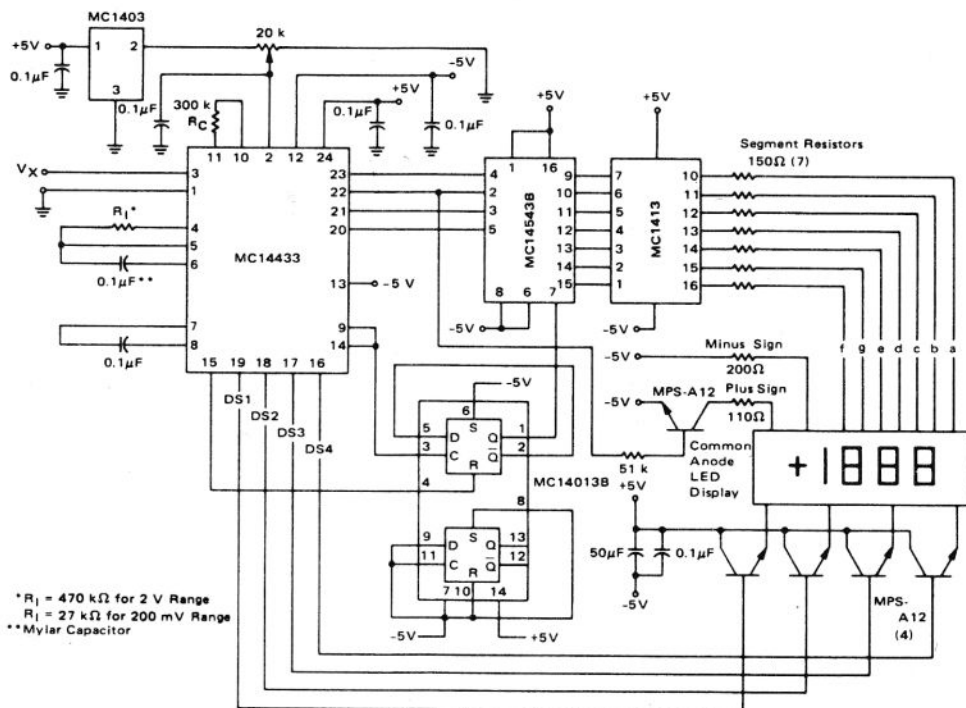
Segment 2 – The integrator output decreases to the comparator threshold voltage. At this time a number of counts equivalent to the input offset voltage of the comparator is stored in the offset latches for later use in the autozero process. The time for this segment is variable, and less than 800 clock periods.

Segment 3 – This segment of the conversion cycle is the same as Segment 1.

Segment 4 – Segment 4 is an up-going ramp cycle with the unknown input voltage (V_X) as the input to the integrator. Figure 10 shows the equivalent configuration of the analog section of the MC14433. The actual configuration of the analog section is dependent upon the polarity of the input voltage during the previous conversion cycle.

Segment 5 – This segment is a down-going ramp period with the reference voltage as the input to the integrator. Segment 5 of the conversion cycle has a time equal to the number of counts stored in the offset storage latches during Segment 2. As a result, the system zeros automatically.

Segment 6 – This is an extension of Segment 5. The time period for this portion is 4000 clock periods. The results of the A/D conversion cycle are determined in this portion of the conversion cycle.

FIGURE 11 – 3-1/2 DIGIT VOLTMETER–COMMON ANODE DISPLAYS, FLASHING OVERRANGE

APPLICATIONS INFORMATION

3½ DIGIT VOLTMETER – COMMON ANODE DISPLAYS, FLASHING OVERRANGE

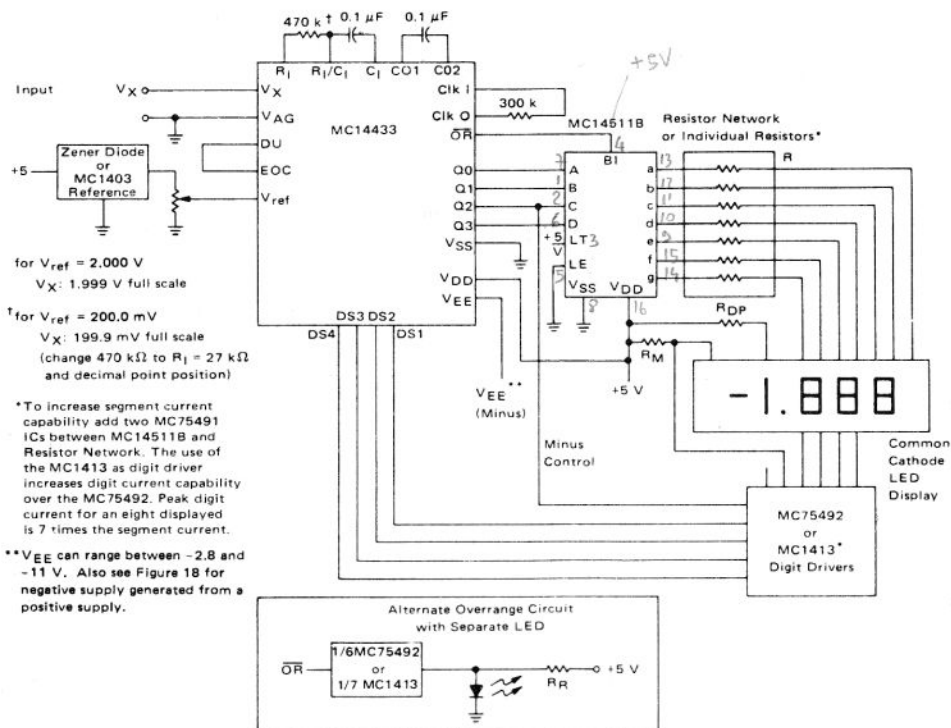
An example of a 3½ digit voltmeter using the MC14433 is shown in the circuit diagram of Figure 11. The reference voltage for the system uses an MC1403 2.5 V reference IC. The full scale potentiometer can calibrate for a full scale of 199.9 mV or 1.999 V. When switching from 2 V to 200 mV operation, R_I is also changed, as shown on the diagram.

When using R_C equal to 300 kΩ, the clock frequency for the system is about 66 kHz. The resulting conversion time is approximately 250 ms.

When the input is overrange, the display flashes on and off. The flashing rate is one-half the conversion rate. This is done by dividing the EOC pulse rate by 2 with ½ MC14013B flip-flop and blanking the display using the blanking input of the MC14543B.

The display uses an LED display with common anode digit lines driven with an MC14543B decoder and an MC1413 LED driver. The MC1413 contains 7 darlington transistor drivers and resistors to drive the segments of the display. The digit drive is provided by four MPS-A12 darlington transistors operating in an emitter follower configuration. The MC14543B, MC14013B and LED displays are referenced to V_{EE} via pin 13 of the MC14433. This places the full power supply voltage across the display. The current for the display may be adjusted by the value of the segment resistors shown as 150 ohms in the above figure.

The power supply for the system is shown as a dual ±5 V supply. However, the MC14433 will operate over a wide range of voltages, and balance between the +5 and –5 V supplies is *not* required. See the recommended operating conditions and Figure 1, on pages 2 and 3.

FIGURE 12 – 3½ DIGIT VOLTMETER WITH LOW COMPONENT COUNT

3½ DIGIT VOLTMETER WITH LOW COMPONENT COUNT USING COMMON CATHODE DISPLAYS

The 3½ digit voltmeter of Figure 12 is an example of the use of the MC14433 in a system with a minimum of components. This circuit uses only 11 components in addition to the MC14433 to operate the MC14433 and drive the LED displays.

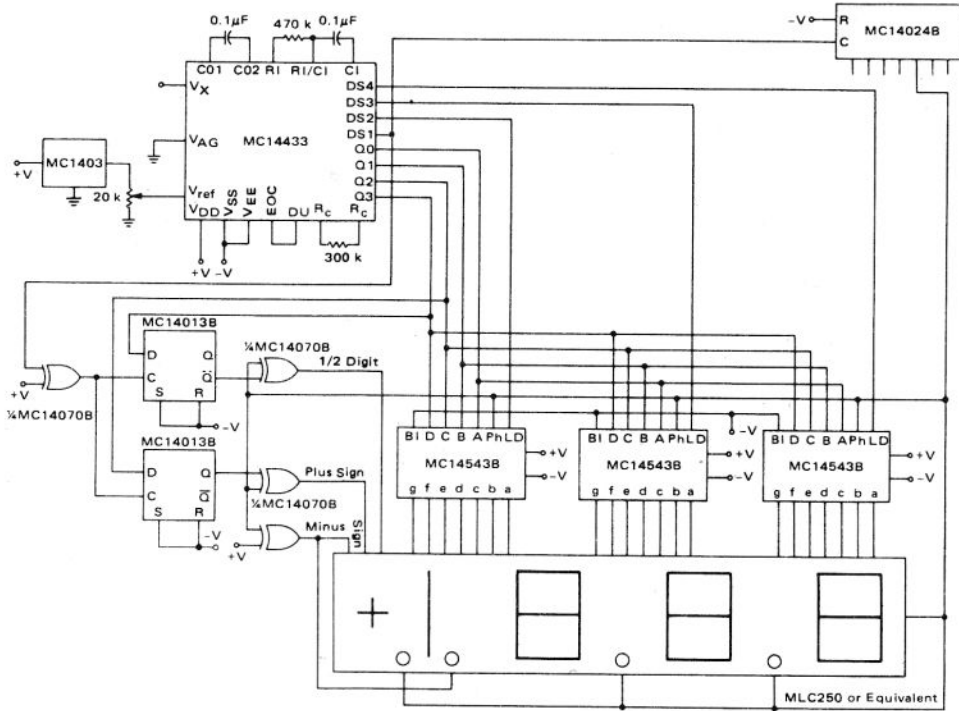
In this circuit the MC14511B provides the segment drive for the 3½ digits. The MC75492 or MC1413 provides sink for digit current. (The MC75492 or MC1413 are devices with 6 or 7 darlingtons respectively with common emitters.) The worst case digit current is 7 times the segment current at ¼ duty cycle. The peak segment current is limited by the value of R. The current for the display flows from V_{DD} (+5 V) to ground and does not flow through the V_{EE} (negative) supply. The minus sign is controlled by one section of the MC75492 or MC1413 and is turned off by shunting the current through R_M to ground, bypassing the minus sign LED. The minus sign is derived from the Q2 output. The decimal point brightness is controlled by resistor R_{DP}. Since the brightness and the type and size of LED display are the choice of the designer, the values of resistors R, R_M, R_{DP}, and R_R that govern brightness are not given.

During an overrange condition the 3½ digit display is blanked at the BI pin on the MC14511B. The decimal point and minus sign will remain on during a negative overrange condition. In addition, an alternate overrange circuit with separate LED is shown. There are leftover sections in either the MC75492 or MC1413.

3½ DIGIT VOLTMETER WITH LCD DISPLAY

A circuit for a 3½ digit voltmeter with a liquid crystal display is shown in Figure 13. Three MC14543B LCD latch/decoder/display drivers are used to demultiplex, decode the three digits, and drive the LCD. The half digit and polarity are demultiplexed with the MC14013B dual D flip-flop.

Since the LCD is best driven by an ac signal across the LCD, the low-frequency square wave drive for the LCD is derived from the MC14024B binary counter which divides the digit select output from the A/D. This low frequency square wave is connected to the backplane of

FIGURE 13 – 3½ DIGIT VOLTMETER WITH LCD DISPLAY

the LCD and to the individual segments through the combination of the output circuitry of the MC14543B and the exclusive OR gates at the outputs of the MC14013B. Alternatively the square wave can be derived from a 50/60 Hz input signal when available.

The minus sign and the decimal point to the right of the half digit are connected to the inverted low frequency square wave signal. Unused decimal points are tied directly to the low frequency square wave.

The system shown operates from two power supplies (plus and minus). Alternatively one supply can be used when V_{SS} is connected to V_{EE}. In this case a level must be set for analog ground, V_{AG}, which must be at least 2.8 V above V_{EE}. This circuit may be implemented with a resistor network, resistor/forward-biased diode network or resistor-zener diode network. For example, a 9 V supply can be used with 3 V between V_{AG} and V_{EE}, leaving 6 V for V_{DD} to V_{AG}. This system leaves a comfortable margin for battery degeneration (end of life). Two versions of this circuit for single supply operation is shown in Figure 14.

For panel meter operation from a single 5 V supply, a negative supply can be generated as shown in Figure 18.

FIGURE 14 – TWO CIRCUITS FOR GENERATION OF V_{ref} AND V_{AG} FROM A SINGLE SUPPLY

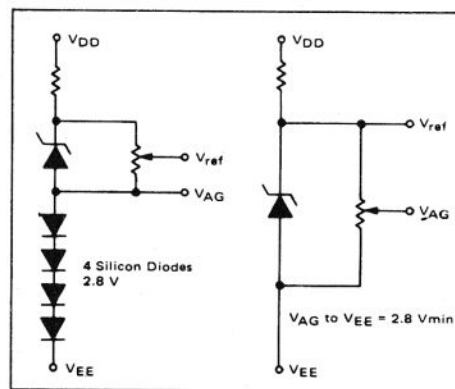
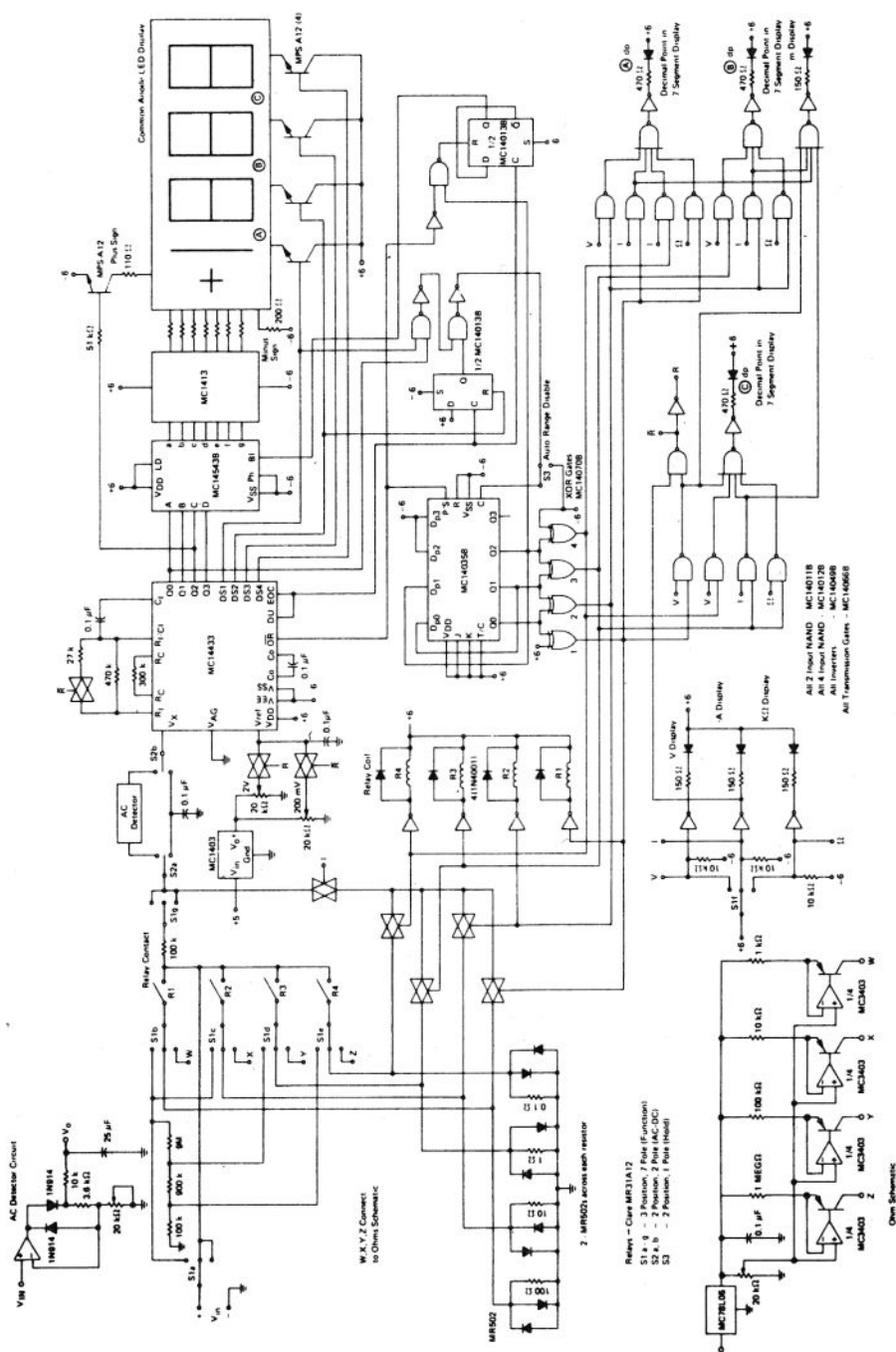



FIGURE 15 – 3½ DIGIT AUTORANGING MULTIMETER

3½ DIGIT AUTORANGING MULTIMETER

An autoranging multimeter including ac and dc voltage ranges from 200 mV to 200 V, ac and dc current from 2 mA to 2 A fullscale and resistance ranges from 2 kΩ to 2 MΩ fullscale is shown in Figure 15. In this multimeter only two input jacks are required for all ranges and functions, eliminating the need for changing leads on the instrument when changing ranges or functions. Although only four ranges are provided for each function, the technique used may be expanded to more ranges if desired. Range switching uses mechanical relays. However, the relays may be replaced with solid state analog switches.

The MC14433 provides the overrange and underrange control signals for the automatic ranging circuits. For additional information, see Motorola Application Note AN-769, "Autoranging Digital Multimeter Using the MC14433 CMOS A/D Converter."

PARALLEL BCD DATA OUTPUT CIRCUIT

The output of the MC14433 may be demultiplexed to produce parallel BCD data as shown in Figure 16. Two levels of latches are required for a complete demultiplexing of the data since the outputs of the MC14042B latches change sequentially with the DS1 to DS4 strobe pulses. To key output validity to one leading edge, i.e., that of the EOC signal of the MC14433, information is transferred to the second set of latches (MC14175B latches). A single set of latches can be used when reading of output is restricted to within 12,000 clock pulses after EOC. This requires synchronous system operation with respect to the BCD data bus.

In this system the output ground level is V_{SS}. In most cases, a two supply system with V_{SS} connected to V_{AG} is recommended. This allows connecting analog ground and digital ground together without destroying a power supply. This circuit works well with that of Figure 12.

FIGURE 16 – DEMULTIPLEXING FOR MC14433 BCD DATA

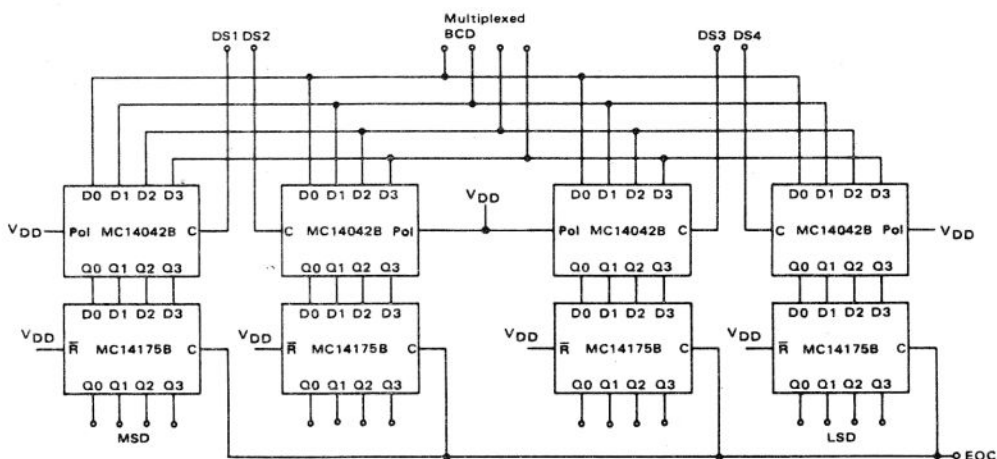



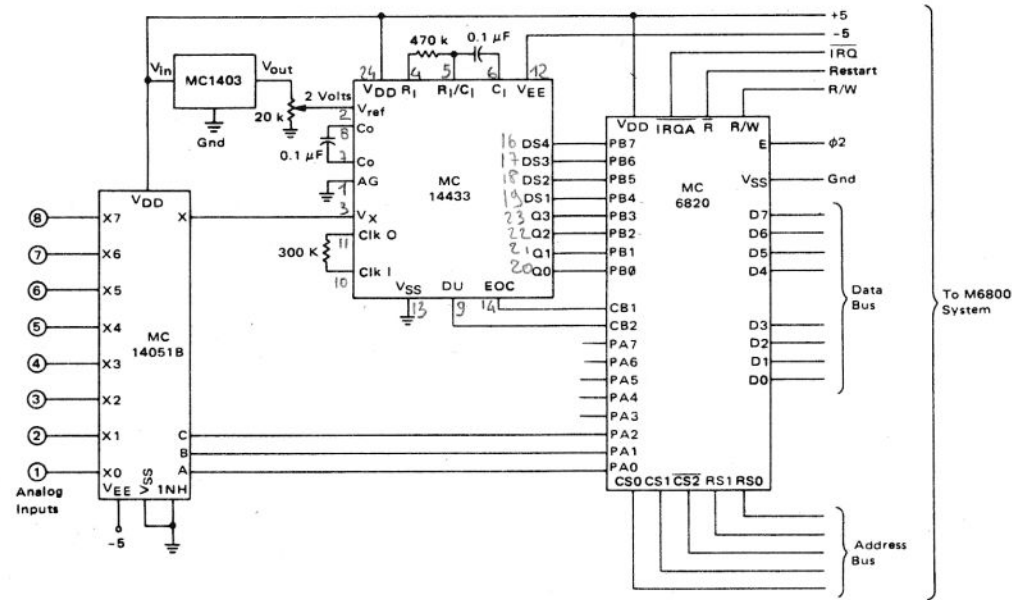
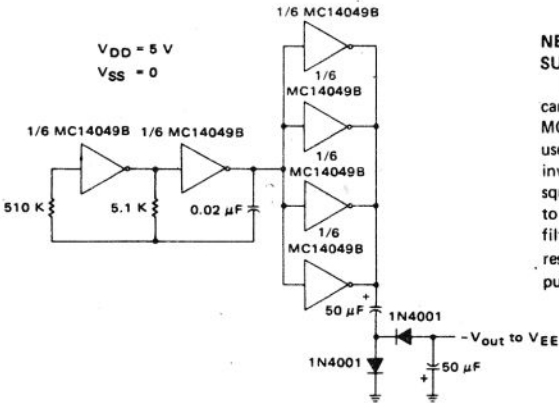
FIGURE 17 – CHANNEL DATA ACQUISITION HARDWARE

8 CHANNEL DATA ACQUISITION NETWORK

Figure 17 shows an 8 channel data acquisition network using the MC14433 and an M6800 microprocessor system. The interface between the microprocessor data bus and the A/D system is done with an MC6820 PIA. One half of the PIA is used with the BCD and digit select outputs of the MC14433, while the second half of the PIA selects the channel to be measured via the MC14051B analog multiplexer. Control lines CB1 and CB2 are used for data flow control and are connected to DU and EOC of the MC14433.

A more detailed explanation of this system including the actual software required for the M6800 microprocessor may be found in Motorola Application Note AN-770, "Data Acquisition Networks With NMOS and CMOS."

FIGURE 18 – NEGATIVE SUPPLY GENERATED FROM POSITIVE SUPPLY

NEGATIVE SUPPLY GENERATED FROM POSITIVE SUPPLY

When only +5 V is available, a negative supply voltage can be generated with the circuit of Figure 18 using one MC14049B. Two inverters from CMOS hex inverter are used as an oscillator (≈ 3 kHz) with the remaining inverters used as buffers for higher current output. The square wave output from the oscillator is level-translated to a negative going signal. This signal is rectified and filtered. A V_{DD} voltage of +5 V for the hex buffer will result in a -4.3 V no load output voltage while the output with a 2 mA load is ≈ 3.4 V.

MC14512B

8-CHANNEL DATA SELECTOR

The MC14512B is an 8-channel data selector constructed with MOS P-channel and N-channel enhancement mode devices in a single monolithic structure. This data selector finds primary application in signal multiplexing functions. It may also be used for data routing, digital signal switching, signal gating, and number sequence generation.

- Quiescent Current = 5.0 nA/package typical @ 5 Vdc
- Noise Immunity = 45% of V_{DD} typical
- Diode Protection on All Inputs
- High Fanout > 50
- Single Supply Operation – Positive or Negative
- 3-State Output (Logic "1", Logic "0", High Impedance)
- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- Capable of Driving Two Low-power TTL Loads, One Low-power Schottky TTL Load or Two HTL Loads Over the Rated Temperature Range

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	-0.5 to +18	Vdc
Input Voltage, All Inputs	V_{in}	-0.5 to V_{DD} + 0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range – AL Device CL/CP Device	T_A	-55 to +125 -40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

TRUTH TABLE

C	B	A	INHIBIT	DISABLE	Z
0	0	0	0	0	X0
0	0	1	0	0	X1
0	1	0	0	0	X2
0	1	1	0	0	X3
1	0	0	0	0	X4
1	0	1	0	0	X5
1	1	0	0	0	X6
1	1	1	0	0	X7
φ	φ	φ	1	0	0
φ	φ	φ	φ	1	High Impedance

φ = Don't Care

CMOS MSI

(LOW-POWER COMPLEMENTARY MOS)

8-CHANNEL DATA SELECTOR

L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

ORDERING INFORMATION

MC14XXXB — Suffix Denotes
- L Ceramic Package
- P Plastic Package
- A Extended Operating Temperature Range
- C Limited Operating Temperature Range

PIN ASSIGNMENT

Pin	Name	Name	Pin
1	X0	V_{DD}	16
2	X1	Dis	15
3	X2	Z	14
4	X3	C	13
5	X4	B	12
6	X5	A	11
7	X6	Inh	10
8	V_{SS}	X7	9

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit.

ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	V_{DD} Vdc	T_{low}* Min	T_{low}* Max	25°C Min	25°C Typ	25°C Max	T_{high}* Min	T_{high}* Max	Unit
Output Voltage "0" Level $V_{in} = V_{DD}$ or 0	V_{OL}	5.0	–	0.05	–	0	0.05	–	0.05	Vdc
		10	–	0.05	–	0	0.05	–	0.05	
		15	–	0.05	–	0	0.05	–	0.05	
"1" Level $V_{in} = 0$ or V_{DD}	V_{OH}	5.0	4.95	–	4.95	5.0	–	4.95	–	Vdc
		10	9.95	–	9.95	10	–	9.95	–	
		15	14.95	–	14.95	15	–	14.95	–	
Input Voltage# "0" Level	V_{IL}									Vdc
(V_O = 4.5 or 0.5 Vdc)		5.0	–	1.5	–	2.25	1.5	–	1.5	
(V_O = 9.0 or 1.0 Vdc)		10	–	3.0	–	4.50	3.0	–	3.0	
(V_O = 13.5 or 1.5 Vdc)		15	–	4.0	–	6.75	4.0	–	4.0	
"1" Level	V_{IH}									Vdc
(V_O = 0.5 or 4.5 Vdc)		5.0	3.5	–	3.5	2.75	–	3.5	–	
(V_O = 1.0 or 9.0 Vdc)		10	7.0	–	7.0	5.50	–	7.0	–	
(V_O = 1.5 or 13.5 Vdc)		15	11.0	–	11.0	8.25	–	11.0	–	
Output Drive Current (AL Device)	I_{OH}									mAdc
(V_{OH} = 2.5 Vdc) Source		5.0	-1.2	–	-1.0	-1.7	–	-0.7	–	
(V_{OH} = 4.6 Vdc)		5.0	-0.25	–	-0.2	-0.36	–	-0.14	–	
(V_{OH} = 9.5 Vdc)		10	-0.62	–	-0.5	-0.9	–	-0.35	–	
(V_{OH} = 13.5 Vdc)		15	-1.8	–	-1.5	-3.5	–	-1.1	–	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	–	0.51	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	–	1.3	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)		15	4.2	–	3.4	8.8	–	2.4	–	
Output Drive Current (CL/CP Device)	I_{OH}									mAdc
(V_{OH} = 2.5 Vdc) Source		5.0	-1.0	–	-0.8	-1.7	–	-0.6	–	
(V_{OH} = 4.6 Vdc)		5.0	-0.2	–	-0.16	-0.36	–	-0.12	–	
(V_{OH} = 9.5 Vdc)		10	-0.5	–	-0.4	-0.9	–	-0.3	–	
(V_{OH} = 13.5 Vdc)		15	-1.4	–	-1.2	-3.5	–	-1.0	–	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.52	–	0.44	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)		10	1.3	–	1.1	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)		15	3.6	–	3.0	8.8	–	2.4	–	
Input Current (AL Device)	I_{in}	15	–	±0.1	–	±0.00001	±0.1	–	±1.0	µAdc
Input Current (CL/CP Device)	I_{in}	15	–	±0.3	–	±0.00001	±0.3	–	±1.0	µAdc
Input Capacitance (V_{in} = 0)	C_{in}	–	–	–	–	5.0	7.5	–	–	pF
Quiescent Current (AL Device) (Per Package)	I_{DD}	5.0	–	5.0	–	0.005	5.0	–	150	µAdc
		10	–	10	–	0.010	10	–	300	
		15	–	20	–	0.015	20	–	600	
Quiescent Current (CL/CP Device) (Per Package)	I_{DD}	5.0	–	20	–	0.005	20	–	150	µAdc
		10	–	40	–	0.010	40	–	300	
		15	–	80	–	0.015	80	–	600	
Total Supply Current**† (Dynamic plus Quiescent, Per Package) (C_L = 50 pF on all outputs, all buffers switching)	I_T	5.0				I_T = (0.8 µA/kHz) f + I_{DD}				µAdc
		10				I_T = (1.6 µA/kHz) f + I_{DD}				
		15				I_T = (2.4 µA/kHz) f + I_{DD}				
Three-State Leakage Current (AL Device)	I_{TL}	15	–	±0.1	–	±0.00001	±0.1	–	±3.0	µAdc
Three-State Leakage Current (CL/CP Device)	I_{TL}	15	–	±1.0	–	±0.00001	±1.0	–	±7.5	µAdc

*T_{low} = -55°C for AL Device, -40°C for CL/CP Device.
T_{high} = +125°C for AL Device, +85°C for CL/CP Device.
#Noise immunity specified for worst-case input combination.
Noise Margin for both "1" and "0" level = 1.0 Vdc min @ V_{DD} = 5.0 Vdc
2.0 Vdc min @ V_{DD} = 10 Vdc
2.5 Vdc min @ V_{DD} = 15 Vdc
†To calculate total supply current at loads other than 50 pF:

$I_T(C_L) = I_T(50\ pF) + 1 \times 10^{-3} (C_L - 50)\ V_{DD} f$

where: I_T is in µA (per package), C_L in pF, V_{DD} in Vdc, and f in kHz is input frequency.
**The formulas given are for the typical characteristics only at 25°C.

SWITCHING CHARACTERISTICS* (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD}	All Types Typ	All Types Max	Unit
Output Rise Time	t_{TLH}				ns
t_{TLH} = (3.0 ns/pF) C_L + 25 ns		5.0	225	360	
t_{TLH} = (1.5 ns/pF) C_L + 12 ns		10	110	180	
t_{TLH} = (1.1 ns/pF) C_L + 8 ns		15	80	130	
Output Fall Time	t_{THL}				ns
t_{THL} = (1.5 ns/pF) C_L + 47 ns		5.0	130	200	
t_{THL} = (0.75 ns/pF) C_L + 24 ns		10	65	100	
t_{THL} = (0.55 ns/pF) C_L + 17 ns		15	50	80	
Turn-Off Delay Time	t_{PLH}				ns
t_{PLH} = (0.9 ns/pF) C_L + 211 ns		5.0	330	650	
t_{PLH} = (0.3 ns/pF) C_L + 70 ns		10	125	250	
t_{PLH} = (0.23 ns/pF) C_L + 54 ns		15	85	170	
Turn-On Delay Time	t_{PHL}				ns
t_{PHL} = (2.7 ns/pF) C_L + 184 ns		5.0	330	650	
t_{PHL} = (0.9 ns/pF C_L + 61 ns		10	125	250	
t_{PHL} = (0.68 ns/pF) C_L + 47 ns		15	85	170	
3-State Output Delay Times "1" or "0" to High Z, and High Z to "1" or "0"	t_{PHZ}, t_{PLZ}, t_{PZH}, t_{PZL}	5.0	60	150	ns
		10	35	100	
		15	30	75	

*The formula given is for the typical characteristics only.

FIGURE 1 — POWER DISSIPATION TEST CIRCUIT AND WAVEFORM

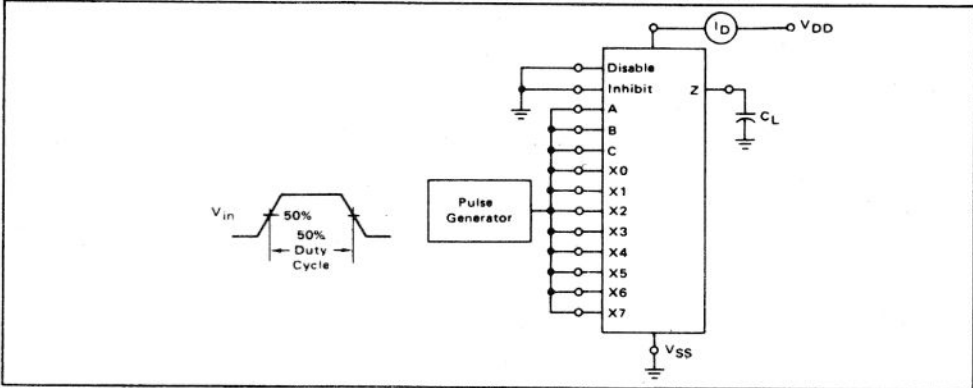



FIGURE 2 — AC TEST CIRCUIT AND WAVEFORMS

TEST	INHIBIT	A	X0
1	P.G.	Gnd	V_{DD}
2	Gnd	P.G.	V_{DD}
3	Gnd	Gnd	P.G.

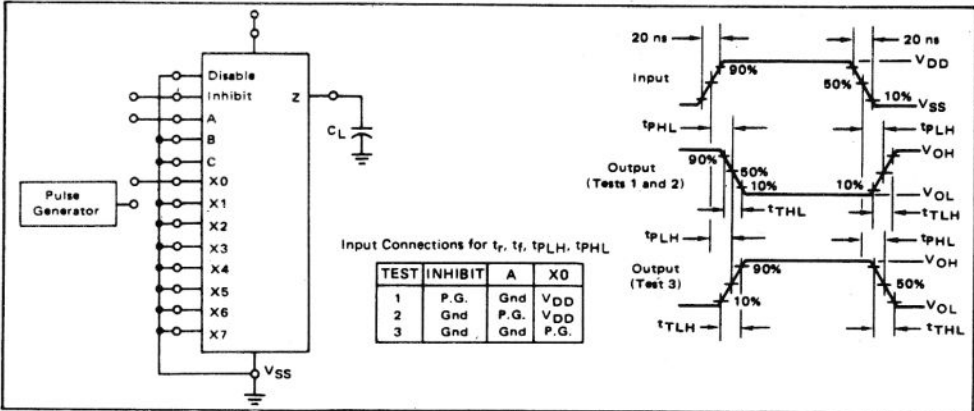



FIGURE 3 – 3-STATE AC TEST CIRCUIT AND WAVEFORM

Switch Positions for 3-State Test

TEST	S1	S2	S3	S4
t_{PHZ}	Open	Closed	Closed	Open
t_{PLZ}	Closed	Open	Open	Closed
t_{PZL}	Closed	Open	Open	Closed
t_{PZH}	Open	Closed	Closed	Open

LOGIC DIAGRAM

3-STATE MODE OF OPERATION

Output terminals of several MC14512B 8-Bit Data Selectors can be connected to a single data bus as shown. One MC14512B is selected by the 3-state control, and the remaining devices are disabled into a high-impedance "off" state. The number of 8-bit data selectors, N, that may be connected to a bus line is determined from the output drive current, I_{OD}, 3-state or disable output leakage current, I_{TL}, and the load current, I_L, required to drive the bus line (including fanout to other device inputs), and can be calculated by:

$$N = \frac{I_{OD} - I_L}{I_{TL}} + 1$$

N must be calculated for both high and low logic state of the bus line.

R65C21
PERIPHERAL INTERFACE ADAPTER (PIA)

PRELIMINARY

DESCRIPTION

The R65C21 Peripheral Interface Adapter (PIA) is designed to solve a broad range of peripheral control problems in the implementation of microcomputer systems. This device allows a very effective trade-off between software and hardware by providing significant capability and flexibility in a low cost chip. When coupled with the power and speed of the R6500, R6500/* or R65C00 family of microprocessors, the R65C21 allows implementation of very complex systems at a minimum overall cost.

Control of peripheral devices is handled primarily through two 8-bit bidirectional ports. Each of these lines can be programmed to act as either an input or an output. In addition, four peripheral control/interrupt input lines are provided. These lines can be used to interrupt the processor or to "handshake" data between the processor and a peripheral device.

ORDERING INFORMATION

The R65C21 is available in both a ceramic and a plastic 40-pin package, a commercial or industrial operating temperature range, and operating frequencies of 1, 2, 3, or 4 MHz. These versions are coded into the part number as follows:

Part Number:
R65C21___

- Temperature Range (T_L to T_H):
  - Blank = 0°C to +70°C
  - E = −40°C to +85°C
- Frequency Range:
  - 1 = 1 MHz
  - 2 = 2 MHz
  - 3 = 3 MHz
  - 4 = 4 MHz
- Package:
  - C = Ceramic
  - P = Plastic

FEATURES

- Low power CMOS N-well silicon gate technology
- Direct replacement for NMOS R6520 or MC6821 PIA
- Two 8-bit bidirectional I/O ports with individual data direction control
- Automatic "Handshake" control of data transfers
- Two interrupts (one for each port) with program control
- 1, 2, 3, and 4 MHz versions
- Commercial and industrial temperature range versions
- 40-pin plastic and ceramic versions
- 5 volt ±5% supply requirements
- Compatible with the R6500, R6500/* and R65C00 family of microprocessors

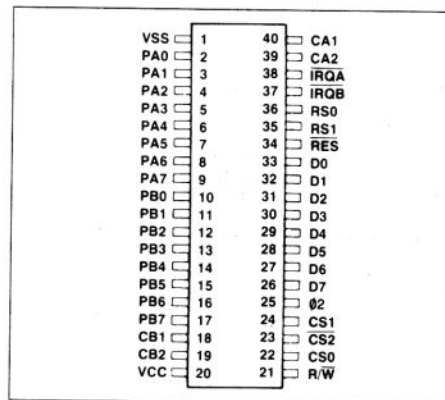



Figure 1. R65C21 Pin Configuration

Document No. 29651N53

Product Description Order No. 2150
Rev. 2, March 1984

FUNCTIONAL DESCRIPTION

The R65C21 PIA is organized into two independent sections referred to as the A Side and the B Side. Each section consists of a Control Register (CRA, CRB), Data Direction Register (DDRA, DDRB), Output Register (ORA, ORB), Interrupt Status Control (ISCA, ISCB), and the buffers necessary to drive the Peripheral Interface buses. Data Bus Buffers (DBB) interface data from the two sections to the data bus, while the Data Input Register (DIR) interfaces data from the DBB to the PIA registers. Chip Select and R/W̄ control circuitry interface to the processor bus control lines. Figure 2 is a block diagram of the R65C21 PIA.

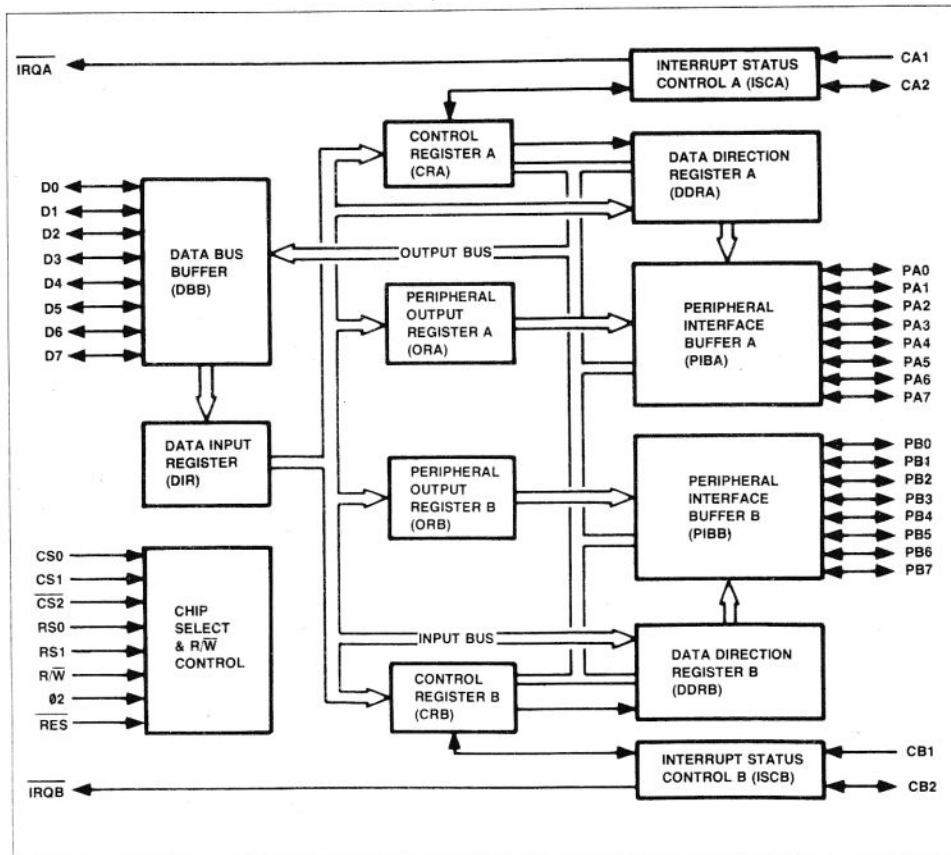



Figure 2. R65C21 PIA Block Diagram

DATA INPUT REGISTER (DIR)

When the microprocessor writes data into the PIA, the data which appears on the data bus during the ø2 clock pulse is latched into the Data Input Register (DIR). The data is then transferred into one of six internal registers of the PIA after the trailing edge of the ø2 clock. This assures that the data on the peripheral output lines will make smooth transitions from high to low (or from low to high) and the voltage will remain stable except when it is going to the opposite polarity.

CONTROL REGISTERS (CRA AND CRB)

Table 1 illustrates the bit designation and functions in the two control registers. The control registers allow the microprocessor to control the operation of the Interrupt Control inputs (CA1, CA2, CB1, CB2), and Peripheral Control outputs (CA2, CB2). Bit 2 in each register controls the addressing of the Data Direction Registers (DDRA, DDRB) and the Output Registers (ORA, ORB). In addition, two bits (bit 6 and 7) in each control register indicate the status of the Interrupt Input lines (CA1, CA2, CB1, CB2). These Interrupt Status bits (IRQA1, IRQA2 or IRQB1, IRQB2) are normally interrogated by the microprocessor during the $\overline{IRQ}$ interrupt service routine to determine the source of the interrupt.

DATA DIRECTION REGISTERS (DDRA, DDRB)

The Data Direction Registers (DDRA, DDRB) allow the processor to program each line in the 8-bit Peripheral I/O port to be either an input or an output. Each bit in DDRA controls the corresponding line in the Peripheral A port and each bit in DDRB controls the corresponding line in the Peripheral B port. Writing a "0" in a bit position in the Data Direction Register causes the corresponding Peripheral I/O line to act as an input; a "1" causes it to act as an output.

Bit 2 (DDRA, DDRB) in each Control Register (CRA and CRB) controls the accessing to the Data Direction Register or the Peripheral interface. If bit 2 is a "1," a Peripheral Output register (ORA, ORB) is selected, and if bit 2 is a "0," a Data Direction Register (DDRA, DDRB) is selected. The Data Direction Register Access Control bit, together with the Register Select lines (RS0, RS1) selects the various internal registers as shown in Table 2.

In order to write data into DDRA, ORA, DDRB, or ORB registers, bit 2 in the proper Control Register must first be set. The desired register may then be accessed with the address determined by the address interconnect technique used.

PERIPHERAL OUTPUT REGISTERS (ORA, ORB)

The Peripheral Output Registers (ORA, ORB) store the output data from the Data Bus Buffers (DBB) which appears on the Peripheral I/O port. If a line on the Peripheral A Port is programmed as an output by the DDRA, writing a 0 into the corresponding bit in the ORA causes that line to go low (<0.4 V); writing a 1 causes the line to go high. The lines of the Peripheral B port are controlled by ORB in the same manner.

2

INTERRUPT STATUS CONTROL (ISCA, ISCB)

The four interrupt/peripheral control lines (CA1, CA2, CB1, CB2) are controlled by the Interrupt Status Control logic (A, B). This logic interprets the contents of the corresponding Control Register and detects active transitions on the interrupt inputs.

PERIPHERAL I/O PORTS (PA0-PA7, PB0-PB7)

The Peripheral A and Peripheral B I/O ports allow the microprocessor to interface to the input lines on the peripheral device by writing data into the Peripheral Output Register. They also allow the processor to interface with the peripheral device output lines by reading the data on the Peripheral Port input lines directly onto the data bus and into the internal registers of the processor.

Each of the Peripheral I/O lines can be programmed to act as an input or an output. This is accomplished by setting a 1 in the corresponding bit in the Data Direction Register for those lines which are to act as outputs. A 0 in a bit of the Data Direction Register causes the corresponding Peripheral I/O lines to act as an input.

The buffers which drive the Peripheral A I/O lines contain "passive" pull-up devices. These pull-up devices are resistive in nature and therefore allow the output voltage to go to VCC for a logic 1. The switches can sink a full 3.2 mA, making these buffers capable of driving two standard TTL loads.

In the input mode, the pull-up devices are still connected to the I/O pin and still supply current to this pin. For this reason, these lines also represent two standard TTL loads in the input mode.

The Peripheral B I/O port duplicates many of the functions of the Peripheral A port. The process of programming these lines to act as an input or an output is similar to the Peripheral A port, as is the effect of reading or writing this port. However, there are several characteristics of the buffers driving these lines which affect their use in peripheral interfacing.

Table 1. Control Registers Bit Designations

	7	6	5	4	3	2	1	0
CRA	IRQA1	IRQA2	CA2 Control			DDRA Access	CA1 Control	
CRB	IRQB1	IRQB2	CB2 Control			DDRB Access	CB1 Control	

The Peripheral B I/O port buffers are push-pull devices i.e., the pull-up devices are switched OFF in the 0 state and ON for a logic 1. Since these pull-ups are active devices, the logic 1 voltage will not go higher than +2.4V.

Another difference between the PA0-PA7 lines and the PB0 through PB7 lines is that they have three-state capability which allows them to enter a high impedance state when programmed to be used as input lines. In addition, data on these lines will be read properly, when programmed as output lines, even if the data signals fall below 2.0 volts for a "high" state or are above 0.8 volts for a "low" state. When programmed as output, each line can drive at least a two TTL load and may also be used as a source of up to 3.2 milliamperes at 1.5 volts to directly drive the base of a transistor switch, such as a Darlington pair.

Because these outputs are designed to drive transistors directly, the output data is read directly from the Peripheral Output Register for those lines programmed to act as inputs.

The final characteristic is the high-impedance input state which is a function of the Peripheral B push-pull buffers. When the Peripheral B I/O lines are programmed to act as inputs, the output buffer enters the high impedance state.

DATA BUS BUFFERS (DBB)

The Data Bus Buffers are 8-bit bidirectional buffers used for data exchange, on the D0-D7 Data Bus, between the microprocessor and the PIA. These buffers are tri-state and are capable of driving a two TTL load (when operating in an output mode) and represent a one TTL load to the microprocessor (when operating in an input mode).

INTERFACE SIGNALS

The PIA interfaces to the R6500, R6500/* or the R65C00 microprocessor family with a reset line, a ø2 clock line, a read/write line, two interrupt request lines, two register select lines, three chip select lines, and an 8-bit bidirectional data bus.

The PIA interfaces to the peripheral devices with four interrupt/control lines and two 8-bit bidirectional data buses.

Figure 1 (on the front page) shows the pin assignments for these interface signals and Figure 3 shows the interface relationship of these signal as they pertain to the CPU and the peripheral devices.

CHIP SELECT (CS0, CS1, $\overline{CS2}$)

The PIA is selected when CS0 and CS1 are high and $\overline{CS2}$ is low. These three chip select lines are normally connected to the processor address lines either directly or through external decoder circuits. When the PIA is selected, data will be transferred between the data lines and PIA registers, and/or peripheral interface lines as determined by the R/$\overline{W}$, RS0, and RS1 lines and the contents of Control Registers A and B.

RESET SIGNAL ($\overline{RES}$)

The Reset ($\overline{RES}$) input initializes the R65C21 PIA. A low signal on the $\overline{RES}$ input causes all internal registers to be cleared.

CLOCK SIGNAL (ø2)

The Phase 2 Clock Signal (ø2) is the system clock that triggers all data transfers between the CPU and the PIA. ø2 is generated by the CPU and is therefore the synchronizing signal between the CPU and the PIA.

READ/WRITE SIGNAL (R/$\overline{W}$)

Read/Write (R/$\overline{W}$) controls the direction of data transfers between the PIA and the data lines associated with the CPU and the peripheral devices. A high on the R/$\overline{W}$ line permits the peripheral devices to transfer data to the CPU from the PIA. A low on the R/$\overline{W}$ line allows data to be transfered from the CPU to the peripheral devices from the PIA.

REGISTER SELECT (RS0, RS1)

The two Register Select lines (RS0, RS1), in conjunction with the Control Registers (CRA, CRB) Data Direction Register access bits (see Table 1, bit 2) select the various R65C21 registers to be accessed by the CPU. RS0 and RS1 are normally connected to the microprocessor (CPU) address output lines. Through control of these lines, the CPU can write directly into the Control

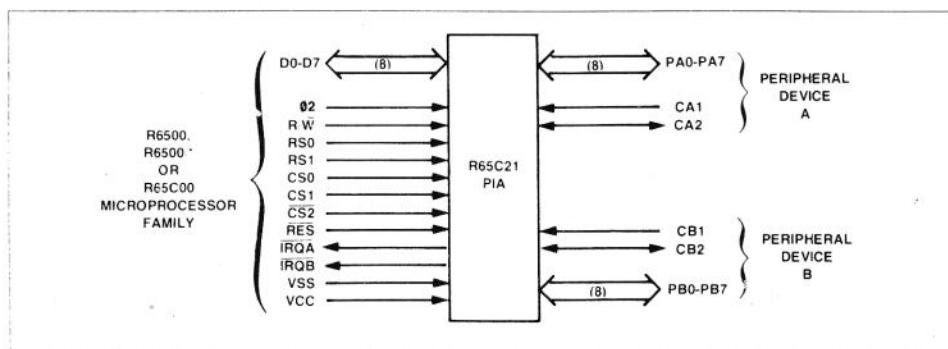



Figure 3. Interface Signals Relationship

Registers (CRA, CRB) the Data Direction Registers (DDRA, DDRB) and the Peripheral Output Registers (ORA, ORB). In addition, the processor may directly read the contents of the Control Registers and the Data Direction Registers. Accessing the Peripheral Output Register for the purpose of reading data back into the processor operates differently on the ORA and the ORB registers and therefore are shown separately in Table 2.

Table 2. ORA and ORB Register Addressing

Register Address (Hex)	Register Select Lines		Data Direction Control		Register Operation	
	RS1	RS0	CRA (Bit 2)	CRB (Bit 2)	R/W̄=H	R/W̄=L
0	L	L	1	—	Read PIBA	Write ORA
0	L	L	0	—	Read DDRA	Write DDRA
1	L	H	—	—	Read CRA	Write CRA
2	H	L	—	1	Read PIBB	Write ORB
2	H	L	—	0	Read DDRB	Write DDRB
3	H	H	—	—	Read CRB	Write CRB

INTERRUPT REQUEST LINES (IRQA, IRQB)

The active low Interrupt Request lines (IRQA and IRQB) act to interrupt the microprocessor either directly or through external interrupt priority circuitry. These lines are open drain and are capable of sinking 1.6 milliamps from an external source. This permits all interrupt request lines to be tied together in a wired-OR configuration. The A and B in the titles of these lines correspond to the peripheral port A and the peripheral port B so that each interrupt request line services one peripheral data port.

Each Interrupt Request line has two interrupt flag bits which can cause the Interrupt Request line to go low. These flags are bits 6 and 7 in the two Control Registers (CRA, CRB). These flags act as the link between the peripheral interrupt signals and the microprocessor interrupt inputs. Each flag has a corresponding interrupt disable bit which allows the processor to enable or disable the interrupt from each of the four interrupt inputs (CA1, CA2, CB1, CB2). The four interrupt flags are set (enabled) by active transitions of the signal on the interrupt input (CA1, CA2, CB1, CB2).

CRA bit 7 (IRQA1) is always set by an active transition of the CA1 interrupt input signal. However, IRQA can be disabled by setting bit 0 in CRA to a 0. Likewise, CRA bit 6 (IRQA2) can be set by an active transition of the CA2 interrupt input signal and IRQA can be disabled by setting bit 3 in CRA to a 0.

Both bit 6 and bit 7 in CRA are reset by a "Read Peripheral Output Register A" operation. This is defined as an operation in which the read/write, proper data direction register and register select signals are provided to allow the processor to read the Peripheral A I/O port. A summary of IRQA control is shown in Table 3.

Control of IRQB is performed in exactly the same manner as that described above for IRQA. Bit 7 in CRB (IRQB1) is set by an active transition on CB1 and IRQB from this flag is controlled by CRB bit 0. Likewise, bit 6 (IRQB2) in CRB is set by an active transition on CB2, and IRQB from this flag is controlled by CRB bit 3.

Also, both bit 6 and bit 7 of CRB are reset by a "Read Peripheral B Output Register" operation. A summary of IRQB control is shown in Table 3.

Table 3. IRQA and IRQB Control Summary

Control Register Bits	Action
CRA-7=1 and CRA-0=1	IRQA goes low (Active)
CRA-6=1 and CRA-3=1	IRQA goes low (Active)
CRB-7=1 and CRB-0=1	IRQB goes low (Active)
CRB-6=1 and CRB-3=1	IRQB goes low (Active)

Note:
The flags act as the link between the peripheral interrupt signals and the processor interrupt inputs. The interrupt disable bits allow the processor to control the interrupt function.

2

INTERRUPT INPUT/PERIPHERAL CONTROL LINES (CA1, CA2, CB1, CB2)

The four interrupt input/peripheral control lines provide a number of special peripheral control functions. These lines greatly enhance the power of the two general purpose interface ports (PA0-PA7, PB0-PB7). Figure 4 summarizes the operation of these control lines.

CA1 is an interrupt input only. An active transition of the signal on this input will set bit 7 of the Control Register A to a logic 1. The active transition can be programmed by setting a "0" in bit 1 of the CRA if the interrupt flag (bit 7 of CRA) is to be set on a negative transition of the CA1 signal or a "1" if it is to be set on a positive transition.

NOTE:

A negative transition is defined as a transition from a high to a low, and a positive transition is defined as a transition from a low to a high voltage.

CA2 can act as a totally independent interrupt or as a peripheral control output. As an input (CRA, bit 5 = 0) it acts to set the interrupt flag, bit 6 of CRA, to a logic 1 on the active transition selected by bit 4 of CRA.

These control register bits and interrupt inputs serve the same basic function as that described above for CA1. The input signal sets the interrupt flag which serves as the link between the peripheral device and the processor interrupt structure. The interrupt disable bit allows the processor to exercise control over the system interrupt.

In the output mode (CRA, bit 5 = 1), CA2 can operate independently to generate a simple pulse each time the microprocessor reads the data on the Peripheral A I/O port. This mode is selected by setting CRA, bit 4 to a 0 and CRA, bit 3 to a 1. This pulse output can be used to control the counters, shift registers, etc., which make sequential data available on the Peripheral input lines.

CONTROL REGISTER A (CRA)

CA2 INPUT MODE (BIT 5 = 0)

7	6	5	4	3	2	1	0
IRQA1 FLAG	IRQA2 FLAG	CA2 INPUT MODE SELECT (=0)	IRQA2 POSITIVE TRANSITION	$\overline{IRQA}$ ENABLE FOR IRQA2	ORA SELECT	IRQA1 POSITIVE TRANSITION	$\overline{IRQA}$ ENABLE FOR IRQA1
			IRQA/IRQA2 CONTROL			$\overline{IRQA}$/IRQA1 CONTROL	

CA2 OUTPUT MODE (BIT 5 = 1)

7	6	5	4	3	2	1	0
IRQA1 FLAG	0	CA2 OUTPUT MODE SELECT (=1)	CA2 OUTPUT CONTROL	CA2 RESTORE CONTROL	ORA SELECT	IRQA1 POSITIVE TRANSITION	$\overline{IRQA}$ ENABLE FOR IRQA1
			CA2 CONTROL			$\overline{IRQA}$/IRQA1 CONTROL	

CA2 INPUT OR OUTPUT MODE (BIT 5 = 0 or 1)

Bit 7 IRQA1 FLAG
- 1 A transition has occurred on CA1 that satisfies the bit 1 IRQA1 transition polarity criteria. This bit is cleared by a read of Output Register A or by $\overline{RES}$.
- 0 No transition has occurred on CA1 that satisfies the bit 1 IRQA1 transition polarity criteria.

Bit 2 OUTPUT REGISTER A SELECT
- 1 Select Output Register A.
- 0 Select Data Direction Register A.

Bit 1 IRQA1 POSITIVE TRANSITION
- 1 Set IRQA1 Flag (bit 7) on a positive (low-to-high) transition of CA1.
- 0 Set IRQA1 Flag (bit 7) on a negative (high-to-low) transition of CA1.

Bit 0 $\overline{IRQA}$ ENABLE FOR IRQA1
- 1 Enable assertion of $\overline{IRQA}$ when IRQA1 Flag (bit 7) is set.
- 0 Disable assertion of $\overline{IRQA}$ when IRQA1 Flag (bit 7) is set.

CA2 INPUT MODE (BIT 5 = 0)

Bit 6 IRQA2 FLAG
- 1 A transition has occurred on CA2 that satisfies the bit 4 IRQA2 transition polarity criteria. This flag is cleared by a read of Output Register A or by $\overline{RES}$.
- 0 No transition has occurred on CA2 that satisfies the bit 4 IRQA2 transition polarity criteria.

Bit 5 CA2 MODE SELECT
- 0 Select CA2 Input Mode.

Bit 4 IRQA2 POSITIVE TRANSITION
- 1 Set IRQA2 Flag (bit 6) on a positive (low-to-high) transition of CA2.
- 0 Set IRQA2 Flag (bit 6) on a negative (high-to-low) transition of CA2.

Bit 3 $\overline{IRQA}$ ENABLE FOR IRQA2
- 1 Enable assertion of $\overline{IRQA}$ when IRQA2 Flag (bit 6) is set.
- 0 Disable assertion of $\overline{IRQA}$ when IRQA2 Flag (bit 6) is set.

CA2 OUTPUT MODE (BIT 5 = 1)

Bit 6 NOT USED
- 0 Always zero.

Bit 5 CA2 MODE SELECT
- 1 Select CA2 Output Mode.

Bit 4 CA2 OUTPUT CONTROL
- 1 CA2 goes low when a zero is written into CRA bit 3. CA2 goes high when a one is written into CRA bit 3.
- 0 CA2 goes low on the first negative (high-to-low) Ø2 clock transition following a read of Output Register A. CA2 returns high as specified by bit 3.

Bit 3 CA2 READ STROBE RESTORE CONTROL (4 = 0)
- 1 CA2 returns high on the next Ø2 clock negative transition following a read of Output Register A.
- 0 CA2 returns high on the next active CA1 transition following a read of Output Register A as specified by bit 1.

Figure 4. Control Line Operations Summary (1 of 2)

CONTROL REGISTER B (CRB)

CB2 INPUT MODE (BIT 5 = 0)

7	6	5	4	3	2	1	0
IRQB1 FLAG	IRQB2 FLAG	CB2 INPUT MODE SELECT (=0)	IRQB2 POSITIVE TRANSITION	$\overline{\text{IRQB}}$ ENABLE FOR IRQB2	ORB SELECT	IRQB1 POSITIVE TRANSITION	$\overline{\text{IRQB}}$ ENABLE FOR IRQB1
			$\overline{\text{IRQB}}$/IRQB2 CONTROL			$\overline{\text{IRQB}}$/IRQB1 CONTROL	

CB2 OUTPUT MODE (BIT 5 = 1)

7	6	5	4	3	2	1	0
IRQB1 FLAG	0	CB2 OUTPUT MODE SELECT (=1)	CB2 OUTPUT CONTROL	CB2 RESTORE CONTROL	ORB SELECT	IRQB1 POSITIVE TRANSITION	$\overline{\text{IRQB}}$ ENABLE FOR IRQB1
			CB2 CONTROL			$\overline{\text{IRQB}}$/IRQB1 CONTROL	

CB2 INPUT OR OUTPUT MODE (BIT 5 = 0 or 1)

Bit 7 IRQB1 FLAG
- 1 A transition has occurred on CB1 that satisfies the bit 1 IRQB1 transition polarity criteria. This bit is cleared by a read of Output Register B or by $\overline{\text{RES}}$.
- 0 No transition has occurred on CB1 that satisfies the bit 1 IRQB1 transition polarity criteria.

Bit 2 OUTPUT REGISTER B SELECT
- 1 Select Output Register B.
- 0 Select Data Direction Register B.

Bit 1 IRQB1 POSITIVE TRANSITION
- 1 Set IRQB1 Flag (bit 7) on a positive (low-to-high) transition of CB1.
- 0 Set IRQB1 Flag (bit 7) on a negative (high-to-low) transition of CB1.

Bit 0 $\overline{\text{IRQB}}$ ENABLE FOR IRQB1
- 1 Enable assertion of $\overline{\text{IRQB}}$ when IRQB1 Flag (bit 7) is set.
- 0 Disable assertion of $\overline{\text{IRQB}}$ when IRQB1 Flag (bit 7) is set.

CB2 INPUT MODE (BIT 5 = 0)

Bit 6 IRQB2 FLAG
- 1 A transition has occurred on CB2 that satisfies the bit 4 IRQB2 transition polarity criteria. This flag is cleared by a read of Output Register B or by $\overline{\text{RES}}$.
- 0 No transition has occurred on CB2 that satisfies the bit 4 IRQB2 transition polarity criteria.

Bit 5 CB2 MODE SELECT
- 0 Select CB2 Input Mode.

Bit 4 IRQB2 POSITIVE TRANSITION
- 1 Set IRQB2 Flag (bit 6) on a positive (low-to-high) transition of CB2.
- 0 Set IRQB2 Flag (bit 6) on a negative (high-to-low) transition of CB2.

Bit 3 $\overline{\text{IRQB}}$ ENABLE FOR IRQB2
- 1 Enable assertion of $\overline{\text{IRQB}}$ when IRQB2 Flag (bit 6) is set.
- 0 Disable assertion of $\overline{\text{IRQB}}$ when IRQB2 Flag (bit 6) is set.

CB2 OUTPUT MODE (BIT 5 = 1)

Bit 6 NOT USED
- 0 Always zero.

Bit 5 CB2 MODE SELECT
- 1 Select CB2 Output Mode.

Bit 4 CB2 OUTPUT CONTROL
- 1 CB2 goes low when a zero is written into CRB bit 3. CB2 goes high when a one is written into CRB bit 3.
- 0 CB2 goes low on the first negative (high-to-low) Ø2 clock transition following a write to Output Register B. CB2 returns high as specified by bit 3.

Bit 3 CB2 WRITE STROBE RESTORE CONTROL (BIT 4 = 0)
- 1 CB2 returns high on the next Ø2 clock negative transition following a write to Output Register B.
- 0 CB2 returns high on the next active CB1 transition following a write to Output Register B as specified by bit 1.

Figure 4. Control Line Operations Summary (2 of 2)

A second output mode allows CA2 to be used in conjunction with CA1 to "handshake" between the processor and the peripheral device. On the A side, this technique allows positive control of data transfers from the peripheral device into the microprocessor. The CA1 input signals the processor that data is available by interrupting the processor. The processor reads the data and sets CA2 low. This signals the peripheral device that it can make new data available.

The final output mode can be selected by setting bit 4 of CRA to a 1. In this mode, CA2 is a simple peripheral control output which can be set high or low by setting bit 3 of CRA to a 1 or a 0 respectively.

CB1 operates as an interrupt input only in the same manner as CA1. Bit 7 of CRB is set by the active transition selected by bit 0 of CRB. Likewise, the CB2 input mode operates exactly the same as the CA2 input modes. The CB2 output modes, CRB bit 5 = 1, differ somewhat from those of CA2. The pulse output occurs when the processor writes data into the Peripheral B Output Register. Also, the "handshaking" operates on data transfers from the processor into the peripheral device.

READING THE PERIPHERAL A I/O PORT

Performing a Read operation with RS1 = 0, RS0 = 0 and the Data Direction Register Access Control bit (CRA-2) = 1, directly transfers the data on the Peripheral A I/O lines to the data bus. In this situation, the data bus will contain both the input and output data. The processor must be programmed to recognize and interpret only those bits which are important to the particular peripheral operation being performed.

Since the processor always reads the Peripheral A I/O port pins instead of the actual Peripheral Output Register (ORA), it is possible for the data read by the processor to differ from the contents of the Peripheral Output Register for an output line. This is true when the I/O pin is not allowed to go to a full +2.4V DC when the Peripheral Output register contains a logic 1. In this case, the processor will read a 0 from the Peripheral A pin, even though the corresponding bit in the Peripheral Output register is a 1.

READING THE PERIPHERAL B I/O PORT

Reading the Peripheral B I/O port yields a combination of input and output data in a manner similar to the Peripheral A port. However, data is read directly from the Peripheral B Output Register (ORB) for those lines programmed to act as outputs. It is therefore possible to load down the Peripheral B Output lines without causing incorrect data to be transferred back to the processor on a Read operation.

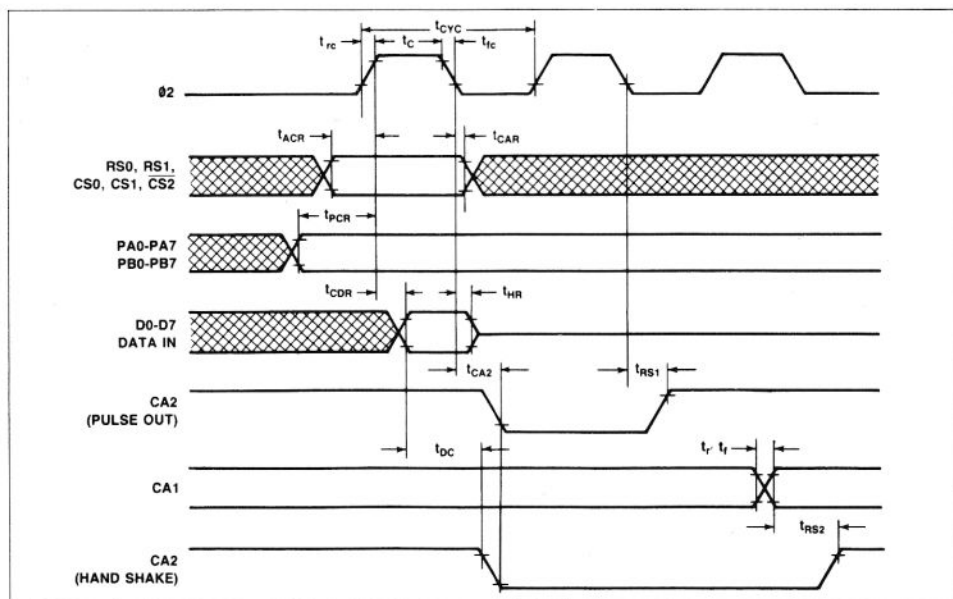



Figure 5. Read Timing Waveforms

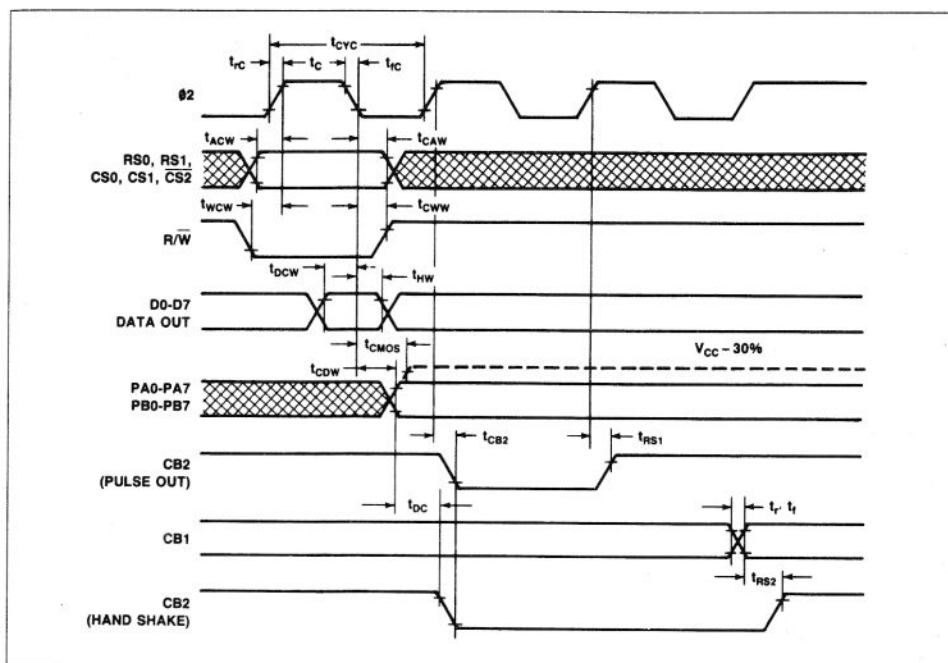



Figure 6. Write Timing Waveforms

BUS TIMING CHARACTERISTICS

Parameter	Symbol	1 MHz		2 MHz		3 MHz		4 MHz		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Ø2 Cycle	t_{CYC}	1.0	—	0.5	—	0.33	—	0.25	—	μs
Ø2 Pulse Width	t_C	480	—	240	—	180	—	120	—	ns
Ø2 Rise and Fall Time	t_{rc}, t_{fc}	—	25	—	15	—	12	—	10	ns

READ TIMING

Parameter	Symbol	1 MHz Min.	1 MHz Max.	2 MHz Min.	2 MHz Max.	3 MHz Min.	3 MHz Max.	4 MHz Min.	4 MHz Max.	Unit
Address Set-Up Time	t_{ACR}	140	—	70	—	53	—	35	—	ns
Address Hold Time	t_{CAR}	0	—	0	—	0	—	0	—	ns
Peripheral Data Set-Up Time	t_{PCR}	300	—	150	—	110	—	75	—	ns
Data Bus Delay Time	t_{CDR}	—	395	—	190	—	100	—	75	ns
Data Bus Hold Time	t_{HR}	20	—	20	—	20	—	20	—	ns

WRITE TIMING

Parameter	Symbol	1 MHz Min.	1 MHz Max.	2 MHz Min.	2 MHz Max.	3 MHz Min.	3 MHz Max.	4 MHz Min.	4 MHz Max.	Unit
Address Set-Up Time	t_{ACW}	140	—	70	—	53	—	35	—	ns
Address Hold Time	t_{CAW}	0	—	0	—	0	—	0	—	ns
R/$\overline{W}$ Set-Up Time	t_{WCW}	180	—	90	—	67	—	45	—	ns
R/$\overline{W}$ Hold Time	t_{CWW}	0	—	0	—	0	—	0	—	ns
Data Bus Set-Up Time	t_{DCW}	180	—	90	—	67	—	45	—	ns
Data Bus Hold Time	t_{HW}	10	—	10	—	10	—	10	—	ns
Peripheral Data Delay Time	t_{CPW}	—	1.0	—	0.5	—	0.33	—	0.25	μs
Peripheral Data Delay Time to CMOS Level	t_{CMOS}	—	2.0	—	1.0	—	0.7	—	0.5	μs

PERIPHERAL INTERFACE TIMING

Parameter	Symbol	1 MHz Min.	1 MHz Max.	2 MHz Min.	2 MHz Max.	3 MHz Min.	3 MHz Max.	4 MHz Min.	4 MHz Max.	Unit
Peripheral Data Set-Up	t_{PCR}	300	—	150	—	110	—	75	—	ns
Ø2 Low to CA2 Low Delay	t_{CA2}	—	1.0	—	0.5	—	0.33	—	0.25	μs
Ø2 Low to CA2 High Delay	t_{RS1}	—	1.0	—	0.5	—	0.33	—	0.25	μs
CA1 Active to CA2 High Delay	t_{RS2}	—	2.0	—	1.0	—	0.67	—	0.5	μs
Ø2 High to CB2 Low Delay	t_{CB2}	—	1.0	—	0.5	—	0.33	—	0.25	μs
Peripheral Data Valid to CB2 Low Delay	t_{DC}	0	1.5	0	0.75	0	0.5	0	0.37	μs
Ø2 High to CB2 High Delay	t_{RS1}	—	1.0	—	0.5	—	0.33	—	0.25	μs
CB1 Active to CB2 High Delay	t_{RS2}	—	2.0	—	1.0	—	0.67	—	0.5	μs
CA1, CA2, CB1 and CB2 Input Rise and Fall Time	t_r, t_f	—	1.0	—	1.0	—	1.0	—	1.0	μs

ABSOLUTE MAXIMUM RATINGS*

Parameter	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	Vdc
Input Voltage	V_{IN}	−0.3 to V_{CC} +0.3	Vdc
Output Voltage	V_{OUT}	−0.3 to V_{CC} +0.3	Vdc
Operating Temperature Range Commercial Industrial	T_A	 0 to +70 −40 to +85	°C
Storage Temperature	T_{STG}	−55 to +150	°C

***NOTE:** Stresses above those listed may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in other sections of this document is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

2

OPERATING CONDITIONS

Parameter	Symbol	Value
Supply Voltage	V_{CC}	5V ±5%
Temperature Range Commercial Industrial	T_A	 0°C to 70°C −40°C to +85°C

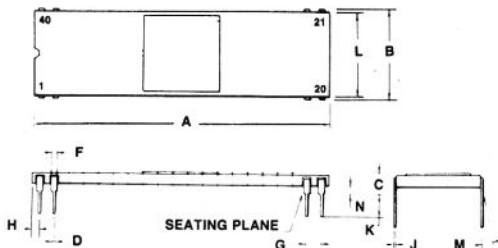
DC CHARACTERISTICS

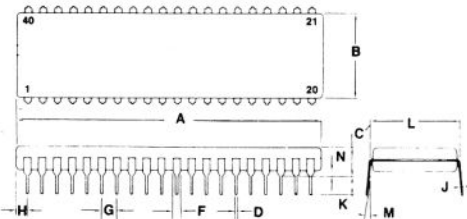
(V_{CC} = 5.0V ±5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted)

Parameter	Symbol	Min.	Typ.[3]	Max.	Unit[2]	Test Conditions
Input High Voltage All except PB0–PB7, $\overline{RES}$ PB0–PB7, $\overline{RES}$	V_{IH}	 +2.0 +2.4	 — —	 V_{CC} V_{CC}	 V V	
Input Low Voltage	V_{IL}	−0.3	—	+0.8	V	
Input Leakage Current R/$\overline{W}$, $\overline{RES}$, RS0, RS1, CS0, CS1, $\overline{CS2}$, CA1, CB1, ø2	I_{IN}	—	±1	±2.5	μA	V_{IN} = 0V to V_{CC} V_{CC} = 5.25V
Input Leakage Current for Three-State Off D0–D7, PB0–PB7, CB2	I_{TSI}	—	±2	±10	μA	V_{IN} = 0.4V to 2.4V V_{CC} = 5.25V
Input High Current PA0–PA7, CA2	I_{IH}	−200	−300	—	μA	V_{IH} = 2.4V
Input Low Current PA0–PA7, CA2	I_{IL}	—	−2	−3.2	mA	V_{IL} = 0.4V
Output High Voltage Logic PB0–PB7, CB2 (Darlington Drive)	V_{OH}	 2.4 1.5	 — —	 — —		V_{CC} = 4.75V I_{LOAD} = −200μA I_{LOAD} = −3.2mA
Output Low Voltage PA0–PA7, CA2, PB0–PB7, CB2 D0–D7, $\overline{IRQA}$, $\overline{IRQB}$	V_{OL}	—	—	+0.4	V	V_{CC} = 4.75V I_{LOAD} = 3.2 mA I_{LOAD} = 1.6 mA
Output High Current (Sourcing Logic PB0–PB7, CB2 (Darlington Drive)	I_{OH}	 −200 −3.2	 −1500 −6	 — —	 μA mA	 V_{OH} = 2.4V V_{OH} = 1.5V
Output Low Current (Sinking) PA0–PA7, PB0–PB7, CB2, CA2 D0–D7, $\overline{IRQA}$, $\overline{IRQB}$	I_{OL}	 3.2 1.6	 — —	 — —	 mA mA	V_{OL} = 0.4V
Output Leakage Current (Off State) $\overline{IRQA}$, $\overline{IRQB}$	I_{OFF}	—	1	±10	μA	V_{OH} = 2.4V V_{CC} = 5.25V
Power Dissipation	P_D		7	10		mW/MHz
Input Capacitance D0–D7, PA0–PA7, PB0–PB7, CA2, CB2 R/$\overline{W}$, $\overline{RES}$, RS0, RS1, CS0, CS1, $\overline{CS2}$ CA1, CB1, ø2	C_{IN}	 — — —	 — — —	 10 7 20	 pF pF pF	V_{CC} = 5.0V V_{IN} = 0V f = 2 MHz T_A = 25°C
Output Capacitance	C_{OUT}	—	—	10	pF	

Notes:
1. All units are direct current (dc) except capacitance.
2. Negative sign indicates outward current flow, positive indicates inward flow.
3. Typical values are shown for V_{CC} = 5.0V and T_A = 25°C.

PACKAGE DIMENSIONS

40-PIN CERAMIC DIP

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	50.29	51.31	1.980	2.020
B	14.86	15.62	0.585	0.615
C	2.54	4.19	0.100	0.165
D	0.38	0.53	0.015	0.021
F	0.76	1.40	0.030	0.055
G	2.54 BSC		0.100 BSC	
H	0.76	1.78	0.030	0.070
J	0.20	0.33	0.008	0.013
K	2.54	4.19	0.100	0.165
L	14.60	15.37	0.575	0.605
M	0	10	0	10
N	0.51	1.52	0.020	0.060

40-PIN PLASTIC DIP

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	51.28	52.32	2.040	2.060
B	13.72	14.22	0.540	0.560
C	3.55	5.08	0.140	0.200
D	0.36	0.51	0.014	0.020
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.30	0.008	0.012
K	3.05	3.56	0.120	0.140
L	15.24 BSC		0.600 BSC	
M	7	10	7	10
N	0.51	1.02	0.020	0.040

OPTOELECTRONICS

Photon Coupled Isolator 4N25-4N25A-4N26-4N27-4N28

Ga As Infrared Emitting Diode & NPN Silicon Photo-Transistor

The General Electric 4N25-4N26-4N27-4N28 consist of a gallium arsenide infrared emitting diode coupled with a silicon photo transistor in a dual in-line package.

FEATURES:

- Fast switching speeds
- High DC current transfer ratio
- High isolation resistance
- 2500 volts isolation voltage
- I/O compatible with integrated circuits

SYMBOL	INCH MIN.	INCH MAX.	MILLIMETER MIN.	MILLIMETER MAX.	NOTES
A	.330	.350	8.38	8.89	
B	.300	REF	7.62	REF	2
C		.340		8.64	3
D	.016	.020	.406	.508	
E		.200		5.08	4
F	.040	.070	1.01	1.78	
G	.090	.110	2.28	2.79	
H		.085		2.16	5
J	.008	.012	.203	.305	
K	.100		2.54		3
M		15°		15°	
N	.015		.381		3
P		.375		9.53	
R	.100	.185	2.54	4.70	
S	.225	.280	5.71	7.12	

NOTES:
1. There shall be a permanent indication of terminal orientation in the quadrant adjacent to terminal 1.
2. Installed position lead centers.
3. Overall installed dimension.
4. These measurements are made from the seating plane.
5. Four places.

†Parameters are JEDEC registered values.

absolute maximum ratings: (25°C) (unless otherwise specified)

†Storage Temperature -55 to 150°C. Operating Temperature -55 to 100°C. Lead Soldering Time (at 260°C) 10 seconds.

INFRARED EMITTING DIODE			PHOTO-TRANSISTOR		
† Power Dissipation	*150	milliwatts	†Power Dissipation	**150	milliwatts
†Forward Current (Continuous)	80	milliamps	†V_{CEO}	30	volts
†Forward Current (Peak) (Pulse width 300 μsec 2% duty cycle)	3	ampere	†V_{CBO}	70	volts
†Reverse Voltage	3	volts	†V_{ECO}	7	volts
			Collector Current (Continuous)	100	milliamps
*Derate 2.0mW/°C above 25°C ambient.			**Derate 2.0mW/°C above 25°C ambient.		

†Total device dissipation @ 24-25°C. P_D 250mW. †Derate 3.3 mW/°C above 25°C ambient.

individual electrical characteristics (25°C)

INFRARED EMITTING DIODE	TYP.	MAX.	UNITS	PHOTO-TRANSISTOR	MIN.	TYP.	MAX.	UNITS
†Forward Voltage (I_F = 10 mA)	1.1	1.5	volts	†Breakdown Voltage – $V_{(BR)CEO}$ (I_C = 1mA, I_F = O)	30	–	–	volts
				†Breakdown Voltage – $V_{(BR)CBO}$ (I_C = 100μA, I_F = O)	70	–	–	volts
†Reverse Current (V_R = 3V)	–	100	microamps	†Breakdown Voltage – $V_{(BR)ECO}$ (I_E = 100μA, I_F = O)	7	–	–	volts
				†Collector Dark Current I_{CEO} 4N25-27 (V_{CE} = 10V, I_F = O)	–	5	50	nanoamps
				4N28	–	–	100	nanoamps
Capacitance V=O,f=1 MHz	50	–	picofarads	†Collector Dark Current – I_{CBO} (V_{CB} = 10V, I_F = O)	–	2	20	nanoamps

coupled electrical characteristics (25°C)

		MIN.	TYP.	MAX.	UNITS
†DC Current Transfer Ratio (I_F = 10mA, V_{CE} = 10V)	4N25, 4N25A, 4N26	20	–	–	%
	4N27, 4N28	10	–	–	%
†Saturation Voltage – Collector – Emitter (I_F = 50mA, I_C = 2 mA)		–	0.1	0.5	volts
Resistance – IRED to Photo-Transistor (@ 500 volts)		–	100	–	gigaohms
Capacitance – IRED to Photo-Transistor (@ 0 volts, f = 1 MHz)		–	1	–	picofarad
†Isolation Voltage – voltage @ 60 Hz with the input terminals (diode) shorted together and the output terminals (transistor) shorted together.	4N25	2500	–	–	volts (peak)
	4N26, 4N27	1500	–	–	volts (peak)
	4N28	500	–	–	volts (peak)
	4N25A	1775	–	–	volts (RMS) (1 sec.)
Rise/Fall Time (V_{CE} = 10V, I_{CE} = 2mA, R_L = 100Ω)		–	2	–	microseconds
Rise/Fall Time (V_{CB} = 10V, I_{CB} = 50μA, R_L = 100Ω)		–	300	–	nanoseconds